UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
ESCOLA DE ENGENHARIA
ENG. DE CONTROLE E AUTOMAÇÃO

FELIPE CARARA

ESTUDO DA UTILIZAÇÃO DO ROS/FLEXBE APLICADO AO CONTROLE DE SISTEMAS A EVENTOS DISCRETOS

Porto Alegre
2019

FELIPE CARARA

ESTUDO DA UTILIZAÇÃO DO ROS/FLEXBE APLICADO AO CONTROLE DE SISTEMAS A EVENTOS DISCRETOS

Trabalho de Conclusão de Curso (TCC-CCA) apresentado à COMGRAD-CCA da Universidade Federal do Rio Grande do Sul como parte dos requisitos para a obtenção do título de *Bacharel em Eng. de Controle e Automação* .

ORIENTADOR: Prof. Dr. Marcelo Götz

Porto Alegre
2019

FELIPE CARARA

ESTUDO DA UTILIZAÇÃO DO ROS/FLEXBE APLICADO AO CONTROLE DE SISTEMAS A EVENTOS DISCRETOS

Este Trabalho de Conclusão de Curso foi julgado adequado para a obtenção dos créditos da Disciplina de TCC do curso *Eng. de Controle e Automação* e aprovado em sua forma final pelo Orientador e pela Banca Examinadora.

Orientador: _____
Prof. Dr. Marcelo Götz, UFRGS
Doutor pela Universität Paderborn, UPB – Paderborn, Alemanha

Banca Examinadora:

Prof. Dr. Alcy Rodolfo dos Santos Carrara, UFRGS
Doutor pela McMaster University – Hamilton, Canadá

Prof. Dr. Marcelo Götz, UFRGS
Doutor pela Universität Paderborn, UPB – Paderborn, Alemanha

Prof. Dr. Renato Ventura Bayan Henriques, UFRGS
Doutor pela Universidade Federal de Minas Gerais – Belo Horizonte, Brasil

_____
Prof. Dr. Marcelo Götz
Coordenador de curso
Eng. de Controle e Automação

Porto Alegre, julho de 2019.

AGRADECIMENTOS

Agradeço à minha família por estar do meu lado em todos os momentos difíceis da faculdade e sempre me proporcionarem um lar cheio de amor e tranquilidade, onde mesmo com todos os momentos difíceis pude me reestruturar e voltar a luta. Agradeço também a todos os almoços em família e todas as datas especiais que me trouxeram muita alegria. Agradeço à minha namorada e sua família, por chegarem em minha vida e constituir minha segunda família, ou melhor do que isso, uma grande família.

Um agradecimento especial à minha mãe, Magda, e ao meu pai, Victorio, por serem a minha base, e sempre me educarem da melhor forma possível, me darem força e sempre me munirem de bons ensinamentos e exemplos.

Ao meu irmão, Bruno, por ser meu irmão mais velho, desbravar o mundo antes de mim e sempre me mostrar os atalhos. Obrigado por ser um espelho para mim.

À minha namorada Débora, por nunca me deixar ficar triste, pensar em desistir e sempre me trazer a sua alegria de viver, me contagiando com seus sorrisos, suas brincadeiras e o seu amor.

À sua vó, Marieta, e dinda, Ângela, por me acolherem e me proporcionarem uma segunda casa, uma extensão da que estive longe, em Taquara, por tanto tempo.

Aos meus amigos e colegas que me proporcionaram diversos momentos bons ao longo desta caminhada e incentivaram a superar alguns problemas e esquecer outros.

Agradeço meu orientador Marcelo Götz, por me ajudar, me incentivar e fazer com que o trabalho fosse mais tranquilo do que imaginei.

Um agradecimento também a toda comunidade do ROS por estar sempre melhorando este sistema que possibilitou este trabalho. Por causa deles também, consegui facilmente encontrar materiais na internet que me possibilitaram aprender sobre todas as ferramentas utilizadas e solucionar todas a dúvidas.

RESUMO

Este trabalho tem como objetivo estudar a aplicação de Controle de Sistemas a Eventos Discretos junto ao Sistema Operacional de Robôs (ROS), posto que este sistema tem a capacidade de integração com diversos softwares de simulação de fábricas, controle de robôs e visão computacional. Em um destes softwares de simulação, o V-REP, será montada uma célula automatizada utilizada em centros de distribuições, e este terá seu comportamento modelado na ferramenta DESTool para geração de um Controle Supervisório. Os supervisórios resultantes, então, serão implementados no software FlexBE e testados no simulador para que se verifique o funcionamento e vantagens da aplicação com o ROS, visando posteriormente implementar em uma fábrica real, minimizando possíveis erros.

Palavras-chave: Engenharia de Controle e Automação, Sistemas de Manufatura Flexíveis, Sistema a Eventos Discretos, Autômatos, Teoria de Controle Supervisório.

ABSTRACT

This present study aims to analyze the application of the Discrete Event System in the Robot Operating System (ROS) due to its ability to integrate several softwares related to factory simulation, robot control and computer vision. In one of these simulation softwares, the V-REP, it will be built an automated cell that will be used in distribution centers whose executions will be modeled by the DESTool software to generate a supervisory control. The resulted supervisories will be implemented in the software FlexBe. Then, they will be tested in the simulator to verify its operation and advantages to be applied with the ROS. By doing so, the goal is to implement the resulted supervisories in a real factory reducing possible mistakes.

Keywords: Control and Automation Engineering, Flexible Manufacturing Systems, Discrete Event System, Automaton, Supervisory Control Theory.

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS

CLP	Computador Lógico Programável
DESTool	Discrete Event Systems Tool
FlexBE	Flexible Behavior Engine
FMS	Sistemas de Manufatura Flexível
ROS	Sistema Operacional de Robôs
SED	Sistemas a Eventos Discretos
V-REP	Virtual Robot Experimentation Platform

1 INTRODUÇÃO

Seja em uma produção de embalagem, seja para montagem de novos equipamentos, nos dias de hoje é cada vez mais exigido maior produtividade e maior redução de custo em linhas de produção. Desta forma, é crescente o número de empresas que buscam realizar projetos de automação de sua produção para atingir tal objetivo. Esta mudança vem mostrando que é possível agregar maior qualidade ao produto fabricado, diminuir o tempo de sua produção e ainda realizar processos que não podem ser executados manualmente (GROOVER, 2011).

O funcionamento destes sistemas automatizados tem por princípio o seu controle por via computadorizada e são modelados como um Sistema a Eventos Discretos. Para isso, utilizam sensores que realizam a coleta de dados dos equipamentos e geram os sinais que envolvem o sistema. Baseado em tais informações, é calculada uma ação de controle que então, acionará ou parará os atuadores e outros *hardwares* incluídos no nível das máquinas.

Para estes sistemas podem ser encontradas na literatura diversas formas de construção e implementação de leis de controle. No presente trabalho, será estudada uma nova forma de implementar a lei que irá reger o funcionamento do sistema, calculada através da metodologia Supervisório Modular Local, proposta por (QUEIROZ; CURY, 2000).

1.1 Motivação

Apesar das grandes vantagens de qualidade e produtividade de uma linha de produção automatizada, é importante ter conhecimento do elevado investimento inicial e empenho da empresa usuária, que em muitos casos são um fator impeditivo para sua construção. Por isso, deve-se realizar um cuidadoso planejamento e projeto de sua operação antes de sua instalação. Para evitar grande parte dos problemas operacionais, de projeto e de configuração do layout da fábrica, a construção de um modelo e a simulação dele é altamente recomendada como técnica de análise quantitativa (GROOVER, 2011).

A simulação é a imitação da operação do processo ou sistema real ao longo do tempo. Ela possibilita o estudo e experimentação de interações internas em um sistema complexo e o conhecimento adquirido durante ela pode ser de grande valor para um sistema ainda em investigação. E com o seu estudo ainda é possível entender qual é a parte do processo que está mais lenta e diminuindo a produtividade, testar novas formas de operação sem interromper a linha já em funcionamento e até mesmo ter o seu tempo acelerado ou desacelerado para um análise mais específica (BANKS, 2005).

1.2 Objetivo

Já bastante difundida, a teoria de controle supervisório possui diversas formas de síntese e implementação. Das propostas direcionadas para aplicação em chão de fábrica, destacam-se as realizadas por (QUEIROZ; CURY, 2002) e (VIEIRA et al., 2017), onde a implementação é realizada em um CLP. Nelas pode ser vista a dificuldade de traduzir os modelos de autômatos do sistema em linguagem *ladder*, que pode ser bastante dificultoso de acordo com o número de estados presentes no modelo e se tornar bastante confuso para manutenção, por não se ter clareza na relação entre registrador e estado/evento.

Desta forma, um dos pontos que deseja-se atingir, é uma forma de implementar a metodologia de Supervisório Modular Local, proposta por (QUEIROZ; CURY, 2000), em uma ferramenta que os modelos sejam visualmente fáceis de compreender e que facilitem a sua manutenção e alteração. Aliado a isso, objetiva-se poder comunicar as máquinas de estados implementadas com ferramentas de simulação para verificação do funcionamento e que posterior a isso, seja fácil de se comunicar com CLPs e possibilitem a instalação em chão de fábrica para a automação de linhas de produção.

1.3 Estrutura do Trabalho

No Capítulo 2 será feita uma revisão da literatura, onde serão revisados conceitos da teoria que será abordada no presente trabalho como Sistemas a Eventos Discretos, Autômatos, Teoria de Controle Supervisório, Sistemas de Manufatura Flexíveis e Sistema Operacional de Robôs. No Capítulo 3 serão explicados os materiais utilizados no desenvolvimento do trabalho, são eles as ferramentas de software DESTool, FlexBE e V-REP. No Capítulo 4 será descrita a metodologia para a estrutura proposta, bem como o passo a passo para sua implementação. Já no Capítulo 5 será apresentado um estudo de caso e resultados que visam validar a metodologia proposta e então, no Capítulo 6 serão apresentadas as conclusões e possíveis trabalhos futuros.

2 REVISÃO DA LITERATURA

2.1 Sistemas a Eventos Discretos

Para a correta compreensão dos sistemas abordados neste trabalho, o primeiro passo passa pela definição do que de fato é um sistema. Para tal, foram encontradas algumas definições na literatura e dentre elas, as que mais se adéquam ao objeto de estudo do presente trabalho podem ser vistas a seguir: "Conjunto de elementos que interagem para a realização de uma função não realizável por nenhum dos elementos individuais" (IEEE. . ., 2000) e "Conjunto de objetos que estão unidos em alguma interação regular ou de interdependência direcionados para o cumprimento de algum propósito" (BANKS, 2005).

Algumas outras definições dos Sistemas a Eventos Discretos (SED), que tem-se interesse, são importantes para a correta distinção aos demais estudados em engenharia, visto que este se trata de um sistema dinâmico e não estático, discreto e não contínuo. Os sistemas estáticos são caracterizados por sua saída depender apenas da entrada dele naquele mesmo instante de tempo. Diferente disso, em um sistema dinâmico, a saída, além da entrada naquele momento, depende também das entradas anteriores, que definem o seu estado naquele momento, possuindo assim, o que pode ser chamado de "memória". E diferente dos sistemas contínuos que seus estados podem assumir qualquer valor no domínio dos números reais, os sistemas discretos são descritos por um conjunto de valores discretos.

Além de serem definidos como um sistema dinâmico discreto, um SED também é caracterizado por ter a evolução de seus estados dirigido por eventos. Como exemplo de eventos, pode ser citado o início e o término de uma tarefa, a chegada de um cliente a uma fila, a recepção de uma mensagem em um sistema de comunicação, dentre outros. Desta forma, é possível apresentar a seguinte definição: SED é um sistema dinâmico que evolui de acordo com a ocorrência abrupta de eventos físicos, em intervalos de tempo em geral irregulares e desconhecidos e que entre a ocorrência de dois eventos consecutivos, o sistema permanece em um estado determinado (CURY, 2001). Do ponto de vista de modelagem, isto implica que se é possível identificar um conjunto de eventos que causam a transição dos estados, então o tempo não é mais considerado como causador da evolução do sistema e não serve mais como variável independente (CASSANDRAS; LAFORTUNE, 2006).

2.1.1 Autômatos

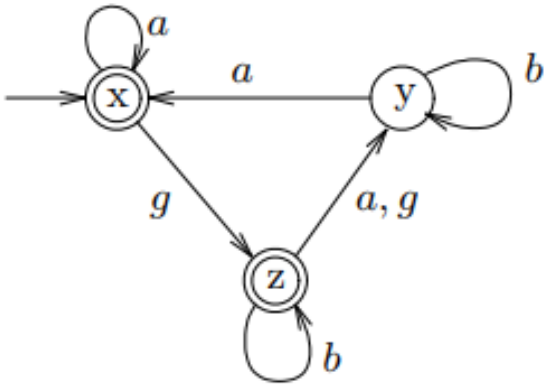
Os autômatos são modelos utilizados para representar Sistemas a Eventos Discretos e considerados uma quíntupla, ou seja, que é definido por cinco argumentos (CURY, 2001). Para explicação desta função, é visto no autômato G a seguir que:

$$G = (X, \sum, f, x_0, X_m)$$

X é o conjunto de todos os estados do autômato, $\sum$ é o conjunto de eventos, f é função de transição, que descreve os estados que são atingidos por cada evento em cada estado, x_0 é o estado inicial e X_m são os estados marcados, isto é, que representam a conclusão de uma atividade.

Um autômato é representado graficamente utilizando a notação de circunferências aos estados, duas circunferências aos estados marcados e aos eventos, flechas que ligam os estados. Na Figura 1, pode ser visto um exemplo de autômato contido em (CURY, 2001):

Figura 1: Exemplo de Autômato.



Fonte: (CURY, 2001)

Onde seus argumentos são:

- $G = (X, \sum, f, x_0, X_m)$
- $X = x, y, z$
- $\sum = a, b, g$
- Funções de transição: $f(x, a) = x, f(x, g) = z, f(y, a) = x, f(y, b) = y, f(z, b) = z, f(z, a) = f(z, g) = y$
- $x_0 = x$
- $X_m = x, z$

2.1.2 Teoria de Controle Supervisório

No comportamento livre do sistema é possível que existam sequências de eventos que não são aceitáveis ou que não são desejadas, podendo se dever à questões de segurança, condições de bloqueio, por violarem um ordenamento desejado ou então por não serem fisicamente admissíveis (CASSANDRAS; LAFORTUNE, 2006). Desta forma, a Teoria de Controle Supervisório concentra-se em definir uma série de restrições de coordenação, que podem vir na forma de especificações modeladas por autômatos, e estas então, são sincronizadas com o autômato do sistema. Assim, de modo a fazer com que o sistema atue na forma desejada, introduz-se um agente de controle denominado supervisor. Este irá atuar de forma a observar a sequência de eventos ocorridos na planta e irá definir então,

quais eventos são permitidos ocorrer, caracterizando um controle de natureza permissiva com uma ação desabilitadora (CURY, 2001).

2.1.3 Controle Monolítico

A metodologia básica para a síntese de um supervisório do tipo Monolítico, proposta inicialmente por Ramadge e Wonham(RW), baseia-se em três passos para gerar um único supervisório que então, irá atuar sobre um modelo único para toda a planta (CURY, 2001). Primeiramente é obtido o modelo da planta, onde faz-se a composição dos subsistemas, isto é, dos modelos das máquinas e obtém-se o comportamento livre dela. O passo seguinte corresponde à identificação de especificações e modelagem destas por autômatos. Via de regra, isso é feito a partir de restrições menores, onde são considerados apenas alguns equipamentos do sistema, e posteriormente calculada uma restrição global com a composição destas.

O último passo é a coordenação com o sistema, fazendo-se a composição das restrições com o modelo da planta, e em seguida a síntese do supervisório que implementa a lógica não bloqueante ótima. Esta se baseia num procedimento iterativo que identifica "maus estados" no autômato e resulta em um comportamento que atende à condição de controlabilidade, que não contém sequências de eventos indesejáveis e que é o menos restritivo possível, garantindo um melhor rendimento.

Como visto acima, um controle monolítico é capaz de resolver problemas de segurança e bloqueio de um sistema em seu comportamento livre. Entretanto, um desafio comum na aplicação prática desta teoria é que o espaço de estados do sistema tende a crescer de uma forma exponencial de acordo com o número de subsistemas e restrições que são combinadas com a composição paralela (CASSANDRAS; LAFORTUNE, 2006). Esta é uma questão importante a se analisar, devido ao fato de para um sistema mais complexo isso exigir uma grande capacidade computacional e um maior tempo de processamento. Desta forma, nas próximas seções serão apresentadas duas alternativas a essa abordagem clássica, introduzindo o conceito de modularidade.

2.1.4 Controle Modular Clássico

Uma das saídas encontradas para contornar esses desafios de complexidade computacional é o Controle Modular Clássico. Nesta metodologia, é utilizada a modularidade das especificações do sistema para então, se construir mais de um supervisório para controlar a planta (CASSANDRAS; LAFORTUNE, 2006).

Nesta abordagem, a ação de controle é decomposta em parcelas menores, onde cada uma será responsabilidade de um dos supervisórios. Para tal, a síntese de cada controlador é dado pela composição da planta completa com determinada especificação ou conjunto de especificações e aplicada a lógica não bloqueante ótima, bem como feito na metodologia anterior. Por fim, é preciso garantir que os comportamentos gerados sejam não conflitantes para que assim, a ação de controle que atue na planta seja resultante da intersecção da ação de controle de cada supervisório. Desta forma, um evento será habilitado, se e somente se, ele estiver presente na ação de ambos controladores.

2.1.5 Controle Modular Local

A abordagem do Controle Modular Local, o qual será utilizado ao longo do presente trabalho, também segue a característica de aproveitar a modularidade das especificações, bem como na metodologia clássica anterior. Esta, entretanto, se difere por não exigir o

cálculo completo da planta, aproveitando também a modularidade de seus subsistemas, e introduzindo assim, o conceito de planta local (QUEIROZ; CURY, 2000).

Para obtenção desta estrutura de supervisório, a primeira etapa é a identificação dos subsistemas, como nas outras metodologias. Seguido do reconhecimento de qual destes são síncronos entre si, isto é, que têm eventos em comum. Isto é importante, pois considera-se um modelo de estrutura descentralizada de operações concorrentes, para um sistema como o que é buscado, aquele que possui o menor número possível de subsistemas síncronos. Assim sendo, ao serem identificado subsistemas que possuem eventos em comum, uma forma de obter esta característica desejada é fazer a composição paralela deles.

O passo seguinte é a identificação de subsistemas assíncronos que compartilham eventos de uma mesma especificação, isto é, todos que são restringidos direta ou indiretamente pela restrição. A composição paralela destes conjuntos de subsistemas assíncronos resulta no que é considerado pela metodologia como planta local. Por fim, utilizando a síntese do supervisório monolítico às plantas locais com suas respectivas especificações, obtém-se o chamado supervisório local.

2.2 Sistema de Manufatura Flexível

Sistemas de Flexíveis de Manufatura normalmente são associados aos princípios da tecnologia de grupo, onde determinados produtos têm processos semelhantes de fabricação e podem ter suas estações de produção agrupadas (GROOVER, 2011). Eles são compostos por grupos de estações de processamento interligados por um sistema automatizado de manuseio e controlado por um sistema que tem a capacidade de identificar diferentes tipos de peças e rearranjar a linha de produção para o correto processamento. Desta forma, com variações nas peças processadas, ele irá automaticamente distingui-las e reorganizar os processos de produção, permitindo obter uma alta diversidade de produtos produzidos.

2.3 Sistema Operacional de Robôs (ROS)

Segundo o livro (O'KANE, 2013), ROS é um pseudo sistema operacional livre e de código aberto para robôs. Ele provê os serviços que são esperados de um sistema operacional, incluindo abstração de hardware, sistema de controle de baixo nível, implementação de funcionalidades comumente usadas, troca de mensagens entre processos e gerenciador de pacotes. Ele também provê ferramentas e bibliotecas para obter, construir, escrever e executar códigos através de múltiplos computadores. No presente trabalho, o ROS será utilizado como meio de comunicação entre o programa onde será implementado o controle supervisório, FlexBE, e o simulador V-REP e isto será feito através dos chamados nodos e tópicos.

Os nodos são instâncias sendo executadas de programas do ROS e são usados para enviar e receber mensagens. Estas mensagens tem seu caminho organizado pelo mecanismo chamado de tópico. A ideia é que um nodo, *publisher*, que quer compartilhar informações irá publicar mensagens em um determinado tópico ou tópicos e o nodo, *subscriber*, que quer receber informações irá assinar um ou mais tópicos que tem interesse. Tal comunicação é gerenciada pelo ROS, garantindo que os *publishers* e os *subscribers* se encontrem e enviem as mensagens diretamente um para o outro.

3 MATERIAIS

3.1 Ferramentas de Software

3.1.1 Destool

DESTool é um ferramenta gráfica que utiliza a biblioteca libFAUDES, a qual implementa estrutura de dados e algoritmos para autômatos finitos e linguagens regulares (MOOR; SCHMIDT; PERK, 2008). Nela é possível fazer a síntese e análise de sistemas a eventos discretos. Com uma interface amigável, é fácil encontrar uma lista de funções, que implementam *scripts* da biblioteca libFAUDES, e que realizam a síntese dos autômatos e controles supervisórios que serão construídos ao longo do trabalho (MOOR, 2019).

3.1.2 FlexBE

FlexBE é uma ferramenta com uma interface de alto nível aplicada ao modelamento de máquinas de estados. Com ele é possível criar comportamentos complexos de uma numerosa quantidade de sistemas e cenários, sem a necessidade de programá-los manualmente (SCHILLINGER; KOHLBRENCHER; VON STRYK, 2016). E o fato de ser criado a partir de pacotes do ROS é um de seus principais diferenciais. Isto permite a criação de tópicos e nodos em sua interface, possibilitando assim, utilizar o ROS como meio de comunicação através do envio de mensagens entre *publishers* e *subscribers*.

3.1.3 V-REP

O simulador de robôs V-REP, possui ambiente integrado de desenvolvimento e é feito com uma arquitetura de controle distribuído. Desta forma, é possível controlar individualmente cada objeto através de um *script* incorporado, *plugin* ou então com o ROS, que será o utilizado no presente trabalho (ROHMER; SINGH; FREESE, 2013). Isso faz do V-REP um ambiente muito versátil para simulação de aplicações com robôs, de sistemas automatizados para fábricas e para rápidas prototipagens e verificações.

4 METODOLOGIA

A metodologia que será descrita a seguir tem o objetivo de poder construir, testar e então implementar um controle supervisório modular local para sistemas a eventos discretos. Desta forma, neste capítulo serão abordados todos os passos utilizados para realização de tal objetivo.

4.1 Modelo e Representação dos Subsistemas no DESTool

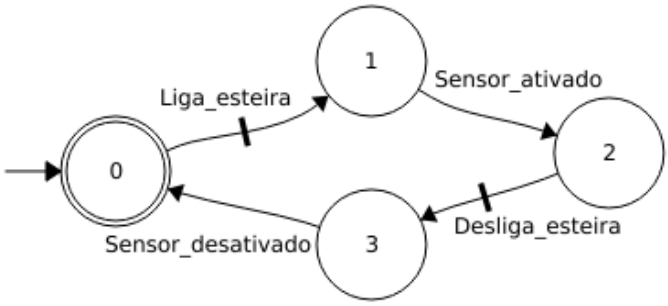
Neste estudo proposto, primeiramente são identificados cada equipamento e seus respectivos estados e eventos. Esta informações serão utilizadas para então, representar estes subsistemas através de autômatos.

Os eventos podem ser do tipo controlável que representarão ações dos equipamentos, como ligar e desligar esteira e acionar ou parar máquina. Ou então podem ser do tipo não controlável que normalmente estão associados à sinais dos sensores, como detecção ou não detecção de presença de alguma peça. Já os estados representarão as situações dos equipamentos de acordo com os eventos e podem ser robô em repouso, robô em movimento ou então presença de caixa ou não na esteira.

Para a construção dos autômatos que modelam cada um dos subsistemas a ferramenta DESTool foi a escolhida e ela também permitirá posteriormente realizar a síntese do controle supervisório que será utilizado. No DESTool isso é feito, na aba “Transitions”, completando as linhas que identificam as transições do sistema. Para isso, é inserido um estado na coluna “X1”, que será representado por uma circunferência, um evento que ocorre neste estado em “Ev”, que será representado por um flecha e esta apontará então, para o segundo estado que deve ser inserido nesta transição, em “X2”. Na Figura 2 pode ser visto um exemplo de autômato de um supervisório representado no DESTool e este será utilizado, a partir daqui, para explicação da metodologia estudada no presente trabalho.

Após a construção dos autômatos que modelam os subsistemas, é importante observar os eventos contidos em cada um deles. Pois para a correta construção do controle modular local desejado, é importante que nenhum dos subsistemas possua eventos em comum. Caso isto ocorra, é recomendado que se faça a composição paralela deles, realizando a combinação dos dois autômatos. Esta tarefa pode ser feita no DESTool utilizando a operação “Parallel”, que é encontrada em “CoreFaudes Operation” na aba “Scripts”. Isso resultará em um novo subsistema que será utilizado na metodologia, para que assim, se obtenha um controle de estrutura descentralizada de operações concorrentes.

Figura 2: Exemplo - Autômato no DESTool.



Fonte: (**Autor**)

4.2 Especificação e Síntese do Supervisório Modular Local

A partir de uma série de especificações de funcionamento do sistema, como comportamento e critérios de desempenho, serão definidos alguns requisitos que irão nortear a atuação do controle supervisório. Tais especificações podem representar sequências de eventos necessárias entre equipamentos, limite máximo de elementos em uma esteira transportadora, ou sequência de eventos que não são desejadas. Estes requisitos do sistema devem ser descritos como pequenas especificações e representadas como autômatos, os quais também serão construídos no DESTool. Chamados de autômatos de restrições, cada um contemplará parte do sistema e irá conter eventos de alguns dos subsistemas modelados anteriormente.

O passo seguinte, que dará início à síntese do supervisório modular local propriamente dita, consiste em identificar os subsistemas que compartilham eventos com cada restrição modelada. Para cada um destes conjuntos deve-se então, ser feita a composição paralela dos subsistemas através da operação "parallel" no DESTool. Esta composição resultará no que é conhecida na literatura como planta local e estas, juntamente com suas respectivas restrições, são chamadas como subsistemas locais. Para a obtenção de um controle modular local, é então, aplicado a cada um destes subsistemas locais a mesma metodologia utilizada na síntese de um supervisório monolítico.

Este método consiste encontrar "maus estados", aplicando um procedimento iterativo que implementa a lógica não bloqueante ótima. Ele fará a síntese do supervisório e irá resultar no autômato menos restritivo possível que atende à restrição imposta e à condição de controlabilidade. No DESTool, isso é possível a partir da utilização da operação de síntese chamada "SupConNB". Mas para que ela funcione e o supervisório resultante tenha a característica permissiva que é buscada, é necessário que as restrições criadas possuam em seus autômatos todos os eventos contidos nos respectivos subsistemas. Estes eventos restantes devem aparecer nos autômatos como eventos que não geram transições, ou seja, como um evento que tem seu destino no mesmo estado de partida. Esta tarefa pode ser feita de forma manual, acrescentando os eventos restantes em cada estado da restrição, ou então por *scripts* do DESTool extraindo os eventos das plantas locais com a função "AlphabetExtract" e então inserindo na restrição os que estão faltando com a operação "InvProject".

Na estrutura mais conhecida de controle supervisório, que pode ser vista em (CURY, 2001), é assumido que a própria planta gera os seus eventos e que o supervisório apenas os observa e desabilita aqueles que não devem ocorrer, de acordo com a sequência de eventos ocorrida. Desta forma, no modelo com o comportamento livre da planta, an-

tes de os eventos ocorrerem, é solicitado ao supervisório sua execução para ver se estes estão habilitados ou não. Entretanto, no modelo proposto no presente trabalho, o próprio supervisório, que contém as cadeias de eventos permitidas, será implementado e será responsável por controlar o sistema, diretamente enviando e recebendo os sinais controláveis e não controláveis dos equipamentos do sistema. Assim, os autômatos que modelam os subsistemas serão utilizados apenas para o cálculo do supervisório, não necessitando a sua implementação e nem de uma interface de comunicação entre eles e o supervisório.

No cálculo do supervisório, a composição das plantas que compartilham eventos de uma mesma restrição e a posterior combinação deles com a respectiva restrição faz com que ocorra o crescimento do número de estados e com que o tamanho do supervisório cresça consideravelmente de acordo o tamanho dos subsistemas. Ao realizar a operação "SupConNB", os eventos que não são permitidos pela restrição ou que podem gerar bloqueio no sistema são removidos, entretanto apenas os eventos controláveis podem ser desabilitados. Então, como não é possível desabilitar eventos não controláveis, caso tenha um deste tipo no início do autômato do subsistema, ele estará presente no supervisório em todas as cadeias geradas pela composição dos subsistemas. Desta forma, para que possa ser gerado um supervisório com tamanho que facilite a sua implementação, é importante estudar a forma que serão modelados os subsistemas. Caso seja possível, é recomendado adiar ao máximo a inserção de um evento não controlável no subsistema. Isto diminuirá o tamanho do supervisório gerado e também pode evitar de acontecer que um supervisório não seja possível de ser calculado, devido às restrições impostas a ele.

4.3 Implementação no FlexBE

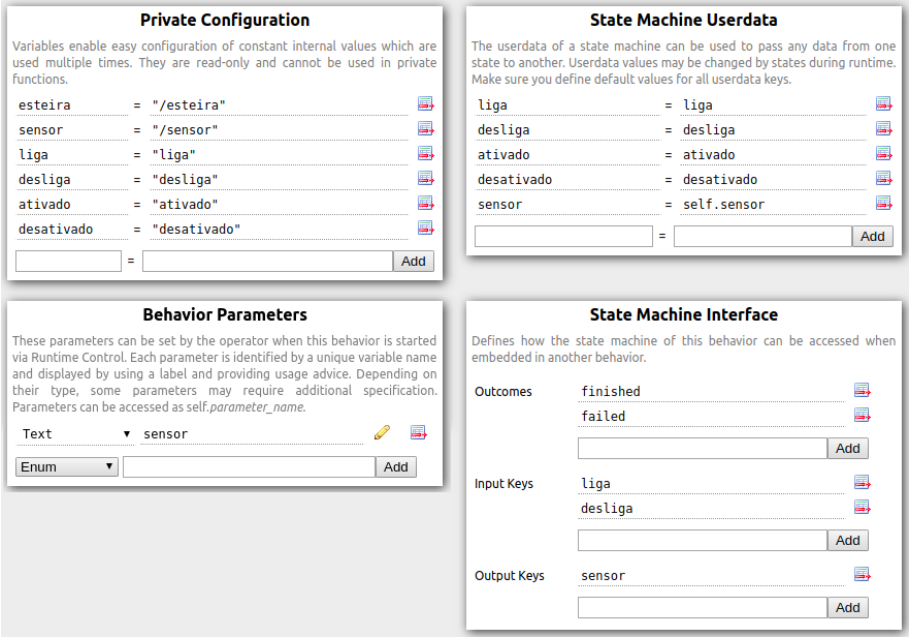
Para se alcançar os objetivos aqui propostos, é estudada uma nova metodologia para implementar os supervisórios. Para tanto, será utilizado o *software* FlexBE, o qual é utilizado para construir máquinas de estado e é feito a partir de pacotes do ROS. Como vantagem desta integração com o ROS, ele permite que os supervisórios ali implementados se comuniquem com o simulador V-REP, que é usado para teste, e esta mesma estrutura pode ser utilizada em uma fábrica real, o qual ainda é visto mais detalhadamente.

Para explicação de como configurar o FlexBE para construção da máquina de estados, é usado o autômato do exemplo anterior, da Figura 2. Na tela chamada de "Behavior Dashboard", que pode ser vista na Figura 3, em "Private Configuration" devem ser criadas as variáveis que irão conter as mensagens e os tópicos por onde são enviadas e recebidas as mensagens. Como no exemplo, onde foi criado "/esteira" e "/sensor", os tópicos sempre são identificados com uma barra seguida de seus respectivos nomes. Já as demais variáveis ali configuradas serão destinadas a comandos da esteira e o estado do sensor.

Em "Behavior Parameters" são criadas as variáveis que terão o seu valor alterado ao longo da execução do programa e no caso do exemplo, será criado uma variável do tipo "texto" que será usada para armazenar a mensagem lida do tópico. Em "State Machine Userdata" essas variáveis então, são instanciadas e em "State Machine Interface" são configurados os comandos que serão enviados nos tópicos em "Input Keys" e as variáveis que serão usadas para armazenamento de mensagem em "Output Keys".

Na tela "State Machine Editor" é onde é criada a máquina de estados. Para que seja possível configurar ela com o mesmo comportamento gerado pelo DESTool, serão usados três tipo de operações do FlexBE. A primeira delas é a "PublisherStringState", Figura 4a, que é responsável por publicar mensagens em um tópico e será usado para implementar os eventos controláveis do sistema. Para isso, deve ser preenchido o campo "topic" com

Figura 3: Configurações - FlexBE.

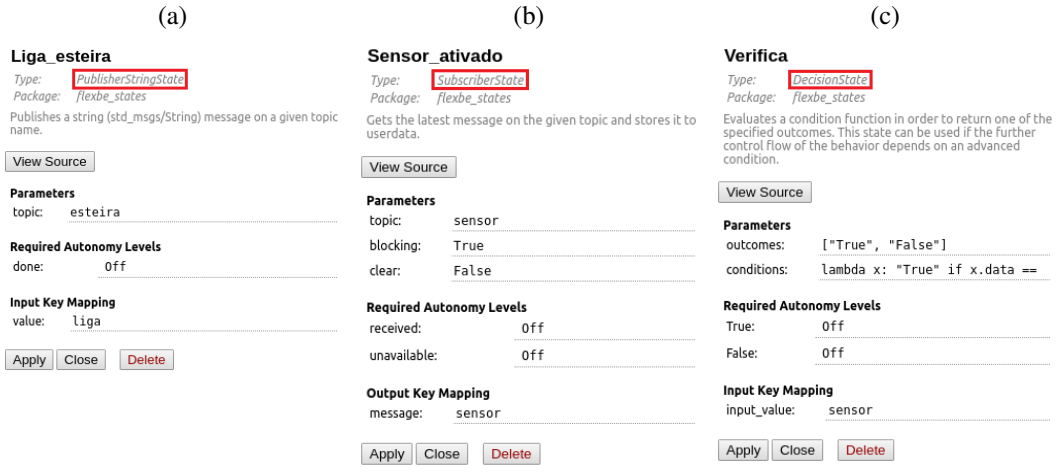


Fonte: (**Autor**)

a variável que contém o nome do tópico de destino e o campo “value” com a variável que armazena a mensagem que será enviada.

Para implementar os eventos não controláveis da planta, primeiramente deve-se adicionar a operação “SubscriberState”, Figura 4b, que lê o tópico especificado em “topic” e armazenar na variável presente em “message”. Em seguida utiliza-se a operação “DecisionState”, Figura 4c para comparar esta mensagem com algum valor de interesse. No exemplo, ela foi configurada podendo ter o resultado verdadeiro ou falso. Esta resposta é dada pela função “lambda x”, que compara o valor da variável sensor, que armazenou a mensagem na operação anterior, com o valor “ativado”, para conferir se o sensor detectou presença de alguma caixa.

Figura 4: Operações - FlexBE.



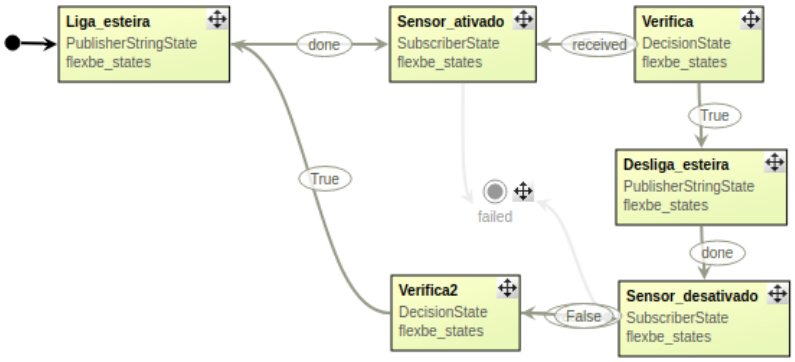
Fonte: (**Autor**)

Com esses três tipos de operação do FlexBE, foram, então, construídos os demais

eventos “desliga_esteira” e “sensor_desativado”. Desta forma, foi montada a máquina de estados que implementa o autômato do exemplo, podendo ser vista na Figura 5. Esses então, são os passos usados para passar os supervisórios gerados pelo DESTool para o FlexBE e na seção “Estudo de Caso e Resultados” é abordado um exemplo completo de uma planta de recebimento de mercadorias para melhor compreensão do método.

É importante observar que na operação “PublisherStringState” são publicadas mensagem do tipo “String”. Entretanto, o FlexBE permite criar novas operações que possibilitem publicar os outros tipos de mensagem. No diretório no qual o FlexBE está configurado encontra-se o código que implementa tal operação e desta forma, com a simples troca da palavra “string” por um outro tipo contido nas mensagens padrões do ROS, “std_msgs/”, como por exemplo “int8”, pode-se criar uma nova operação.

Figura 5: Supervisório no FlexBE.

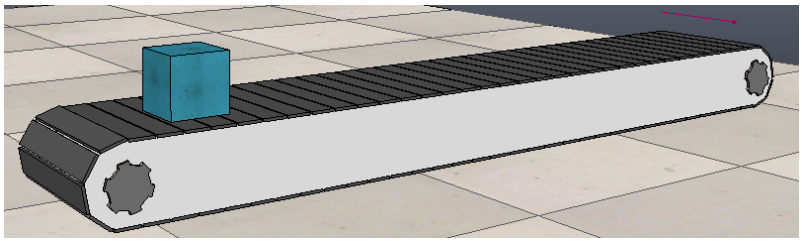


Fonte: (**Autor**)

4.4 Comunicação com o V-REP

Como forma de auxílio na montagem um novo modelo de fábrica ou linha de produção no software V-REP, é possível encontrar alguns cenários de exemplo dentro do próprio diretório o qual ele foi instalado. Para a montagem do exemplo que vem sendo trabalhado, foi adicionado ao cenário inicial uma esteira do tipo “conveyor belt (efficient)”, um sensor de proximidade do tipo “Proximity sensor - Ray”, no canto superior direito da Figura 6, e uma caixa representada por um cubo. Tanto o conjunto esteira com o sensor de proximidade quanto o cubo podem ser encontrados no cenário de exemplo “barrettHand-PickAndPlace.ttt”.

Figura 6: V-REP.



Fonte: (**Autor**)

Utilizando-se a funcionalidade do simulador, de poder controlar individualmente cada objeto atribuindo-o um *script*, é programado um código na esteira para ler e publicar nos

tópicos criados anteriormente. Para isso, foi adicionado à função inicial dela "sysCall_init" as seguintes duas linhas de código:

```
subscriber_esteira=simROS.subscribe('/esteira','std_msgs/String',
'subscriber_esteira_callback')
publisher_sensor=simROS.advertise('/sensor', 'std_msgs/String')
```

A primeira linha cria um *subscriber*, que é responsável por ler as mensagens do tópico "/esteira". Com ele, sempre que uma nova mensagem for recebida, é chamada a função "subscriber_esteira_callback" que então, gera uma ação na planta de acordo com o conteúdo da mensagem. Já a segunda linha cria um *publisher*, que publica mensagens do tipo "String" no tópico "/sensor", o qual é lido no autômato implementado no FlexBE. Tal funcionalidade permitirá ao sensor informar ao supervisório a alteração de seu estado. Desta forma, sempre que uma nova mensagem precisar ser publicada, deve-se utilizar a instrução abaixo, onde é identificado o *publisher* e o conteúdo da mensagem.

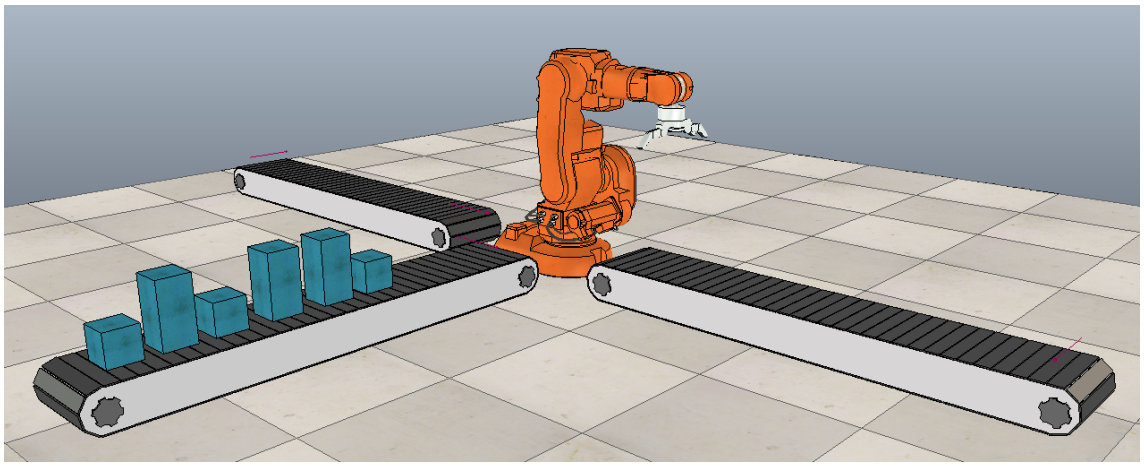
```
simROS.publish(PUBLISHER, {data="MENSAGEM"})
```

Vale ressaltar que nessa última parte da metodologia, é feita uma interface de comunicação entre o FlexBE e o V-REP utilizando o ROS. Isto significa que é possível implementar tal método em qualquer software simulador que permita comunicação com o ROS, através de *publishers* e *subscribers*. Além disso, é abordado posteriormente a existência de uma interface entre ROS e CLP. Isso permite, sem alterar em nada o supervisório implementado no FlexBE, enviar e receber os comandos entre computador e CLP, e desta forma permite a implementação da automação de uma fábrica através de autômatos.

5 ESTUDO DE CASO E RESULTADOS

Para validação da metodologia proposta, foi montada no V-REP a planta da Figura 7. Ela é constituída de uma esteira central e duas esteiras laterais do tipo “conveyor belt (efficient)”, caixas de dois diferentes tamanhos representadas por cubos e paralelepípedos azuis, sensores de presença do tipo “Proximity sensor - Ray” nas extremidades das esteiras e um robô ABB IRB 140 com uma garra do tipo “Barret hand”. Tanto o conjunto das esteiras com o sensor na extremidade quanto o robô com a garra podem ser encontrados no cenário de exemplo, disponibilizado no diretório que o V-REP foi instalado com o nome “barrettHandPickAndPlace.ttt”, o mesmo utilizado no exemplo do Capítulo 4.

Figura 7: Planta - V-REP.



Fonte: (**Autor**)

5.1 Funcionamento do Sistema e de seus Componentes

O funcionamento do sistema é dado pela entrada de novas caixas a partir da esteira central de forma manual, acrescentando novas caixas no simulador, o que no caso real pode ser a partir do descarregamento de mercadorias de um caminhão. Esta esteira transportará as caixas até a sua extremidade, onde haverão dois sensores ópticos de presença que irão detectar a chegada de caixas para então, parar a esteira. Os sensores estarão em diferentes alturas e são responsáveis por identificar os dois diferentes tipos de caixa.

O robô então, recebe esta informação de tamanho da caixa, e é comandado para pegá-la em uma posição determinada no final da esteira central e em seguida, levar para a esteira da esquerda, caso ela for grande ou então para a da direita, caso for pequena. No final das esteiras laterais também existem sensores de presença que param as esteiras para seu

devido descarregamento.

Para dar início à modelagem do comportamento de cada equipamento do sistema, é preciso primeiramente conhecer todos os eventos que podem ocorrer durante o seu funcionamento. Desta forma, estes foram identificados e podem ser vistos na Tabela 1, assim como o componente responsável por eles e a sua controlabilidade.

Tabela 1: Eventos contidos no sistema.

Evento	Componente	Controlabilidade
Liga_esteira1	Esteira central	Controlável
alto_sensor1	Esteira central	Não controlável
Baixo_sensor1	Esteira central	Não controlável
Desliga_esteira1	Esteira central	Controlável
False_sensor1	Esteira central	Não controlável
Direita_robo	Robô	Controlável
Esquerda_robo	Robô	Controlável
Centro_robo	Robô	Controlável
Liga_esteira2	Esteira - Direita	Controlável
True_sensor2	Esteira - Direita	Não controlável
Desliga_esteira2	Esteira - Direita	Controlável
False_sensor2	Esteira - Direita	Não controlável
Liga_esteira3	Esteira - Esquerda	Controlável
True_sensor3	Esteira - Esquerda	Não controlável
Desliga_esteira3	Esteira - Esquerda	Controlável
False_sensor3	Esteira - Esquerda	Não controlável

Fonte: Autor.

Com base nos eventos identificados, deve-se então, descrever o funcionamento de cada máquina do sistema com a finalidade de poder montar os autômatos que representam os seus comportamentos. Estes são descritos da seguinte forma:

- Esteira central: Partindo do princípio que esta já inicia com caixas em sua superfície, o seu primeiro evento será para ligá-la. Então, ao chegar nos sensores de presença, é recebido o sinal de caixa grande ou pequena e a desliga. Quando o sensor detectar que não há mais caixa na extremidade, devido à retirada delas pelo robô manipulador, ela é novamente ligada, recomeçando o seu comportamento cíclico.

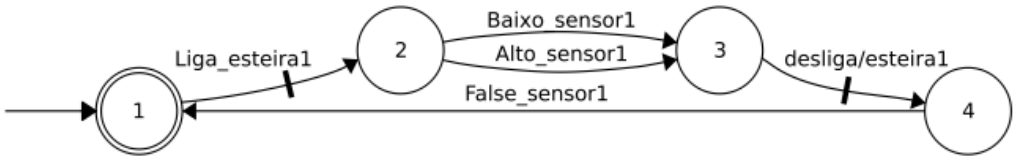
- Robô: Começando em sua posição de repouso, o robô pode receber o comando "Direita_robo" ou "Esquerda_robo" para que então, ele pegue as caixas na posição determinada no final da esteira central e leve para a respectiva esteira lateral. Ao término da tarefa, ele está apto a receber o comando para voltar à posição central e então para buscar novas caixas.

- Esteira - Esquerda: Ela começa sendo ligada e então, irá transportar as caixas até a sua outra extremidade, onde está o sensor de presença. Este detecta-a, e manda um comando "True_sensor2" que então, desliga a esteira. Quando a caixa for retirada, o sensor envia o comando "False_sensor2" que liga novamente a esteira.

- Esteira - Direita: O funcionamento desta esteira é similar ao da esteira anterior.

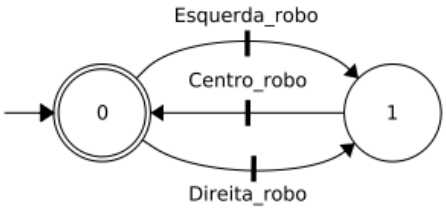
Os autômatos que modelam o funcionamento dos componentes descritos acima podem ser vistos nas figuras 8, 9, 10 e 11.

Figura 8: Autômato da esteira 1 - DESTool.



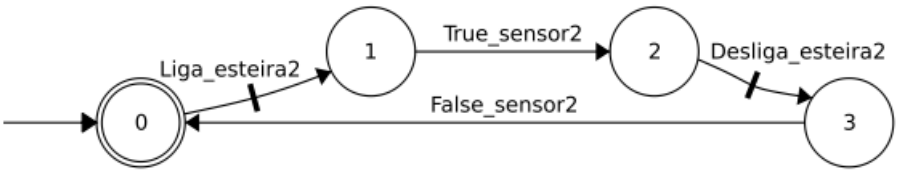
Fonte: (**Autor**)

Figura 9: Autômato do robô - DESTool.



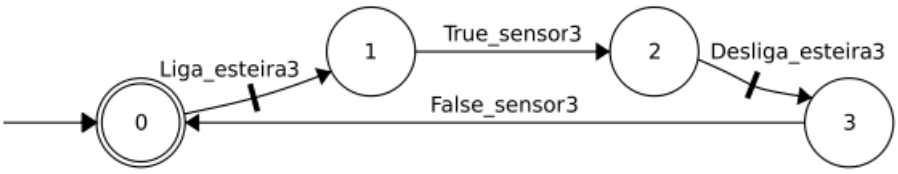
Fonte: (**Autor**)

Figura 10: Autômato da esteira da direita - DESTool.



Fonte: (**Autor**)

Figura 11: Autômato da esteira da esquerda - DESTool.



Fonte: (**Autor**)

5.2 Especificações do sistema

O primeiro passo para a síntese do controle supervisório modular local passa por identificar no sistema restrições que especifiquem as interações entre os equipamentos, representando uma sequência de operações que é executada. Estas restrições contarão com os eventos de cada componente, descritos anteriormente, e são modeladas da seguinte forma:

- Especificação 1: Modela a interação entre a esteira central e o robô manipulador. Na esteira central, quando as caixas transportadas chegarem à sua extremidade, estas

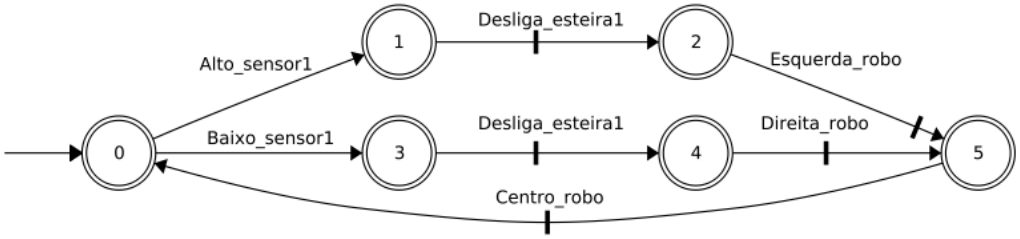
são detectadas pelos sensores de presença. Caso os dois sensores sejam acionados, a caixa é identificada como grande. Entretanto, caso apenas o sensor de baixo detecte presença de algo, ela é identificada como caixa pequena. Em ambas as situações, é requisitada a parada da esteira e então enviado o comando para o robô pegar a caixa e levar para a esteira da direita, se ela for pequena, ou então para a da esquerda, caso ela seja grande. Após o término desta operação, o robô retornará à sua posição central, onde estará pronto para buscar novas caixas.

- Especificação 2: Esta restrição modelará o funcionamento entre o robô manipulador e a esteira da direita. A sequência contida nele consiste em ligar a esteira da direita quando o robô receber o comando "direita_robô" e permitir que ela seja desligada depois que o robô seja mandado para sua posição central.

- Especificação 3: Especifica a interação entre o robô manipulador e a esteira da esquerda e tem a mesma forma da especificação 2, descrita anteriormente, com a diferença de a esteira da esquerda ser ligada, quando o robô receber o comando "esquerda_robô".

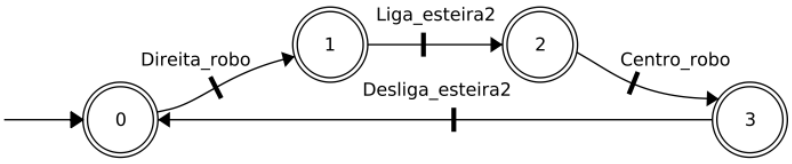
As especificações descritas acima modeladas em autômatos podem ser vistas nas figuras 12, 13 e 14.

Figura 12: Autômato da restrição 1 - DESTool.



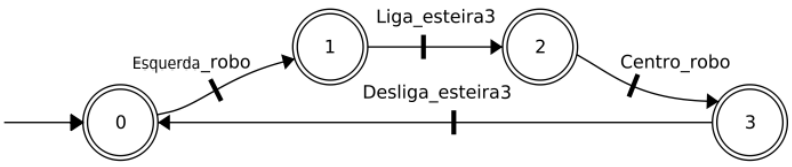
Fonte: (**Autor**)

Figura 13: Autômato da restrição 2 - DESTool.



Fonte: (**Autor**)

Figura 14: Autômato da restrição 3 - DESTool.



Fonte: (**Autor**)

5.3 Cálculo dos supervisórios

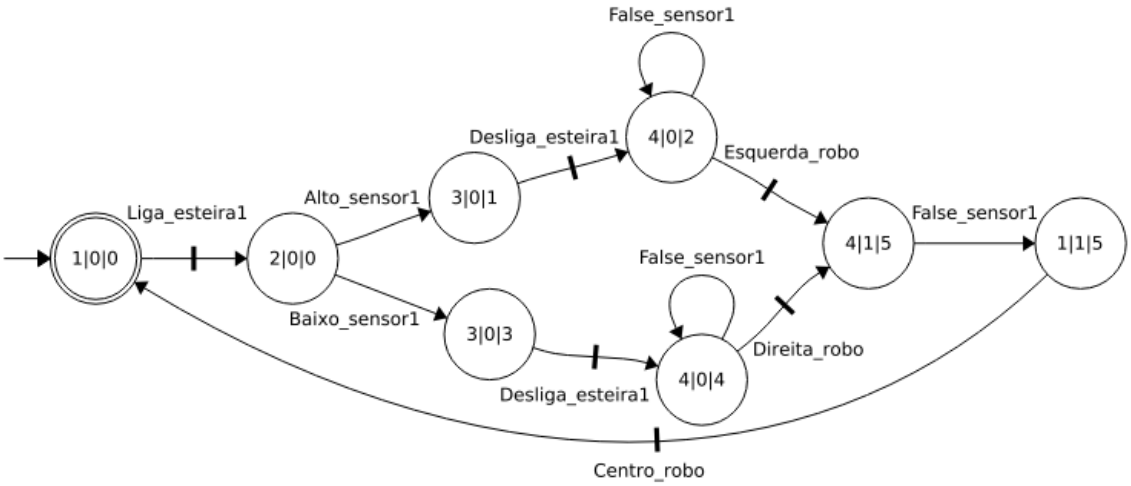
Seguindo a metodologia de controle supervisório modular local, será calculado um controlador para cada restrição, juntamente com os subsistemas assíncronos da planta que possuem eventos em comum a ela. Desta forma, para o controle da planta completa estudada, serão calculados três controles supervisórios que irão gerenciar as operações de cada subsistema e assim, chegar ao comportamento esperado do funcionamento automático de recebimento das novas mercadorias. Estes conjuntos, chamados de subsistemas locais, serão estabelecidos da seguinte forma no caso estudado:

- Subsistema Local 1: Especificação 1 + Esteira central + Robô manipulador
- Subsistema Local 2: Especificação 2 + Robô manipulador + Esteira - Direita
- Subsistema Local 3: Especificação 3 + Robô manipulador + Esteira - Esquerda

Para o cálculo do supervisório na ferramenta DESTool, primeiramente será feita a composição paralela dos subsistemas de cada planta local. Isso é feito utilizando a operação "Parallel", que é encontrado em "CoreFaudes Operation", na aba "Scripts". O próximo passo é o cálculo do controle ótimo menos restritivo possível, que então gerará o supervisório que será implementado. No DESTool, isso é possível a partir da utilização da operação de síntese chamada "SupConNB", utilizando como argumento o resultado da composição dos subsistemas e a respectiva especificação. Mas para que ele funcione e o supervisório resultante tenha a característica permissiva que é buscada, é necessário que as restrições criadas possuam em seus autômatos todos os eventos contidos nos respectivos subsistemas.

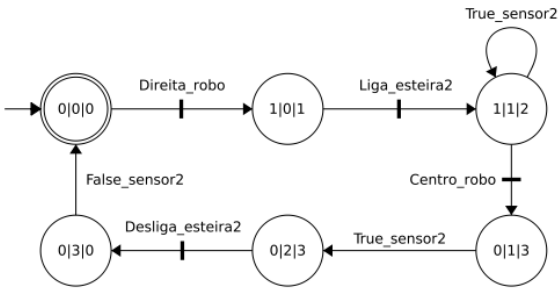
Estes eventos restantes devem aparecer nos autômatos como eventos que não geram transições, ou seja, como um evento que tem seu destino no mesmo estado de partida. Esta tarefa pode ser feita de forma manual, acrescentando os eventos restantes em cada estado da restrição, ou então por *scripts* do DESTool extraindo os eventos dos subsistemas compostos com a função "AlphabetExtract" e então inserindo na restrição os que estão faltando com a operação "InvProject". Os supervisórios 1 e 2 gerados pode ser vistos nas figuras 15 e 16. O supervisório 3 possui a mesma estrutura do supervisório 2, com a diferença de possuir os eventos referentes à esteira 3 e não à esteira 2.

Figura 15: Autômato supervisório 1 - DESTool.



Fonte: (**Autor**)

Figura 16: Autômato supervisório 2 - DESTool.



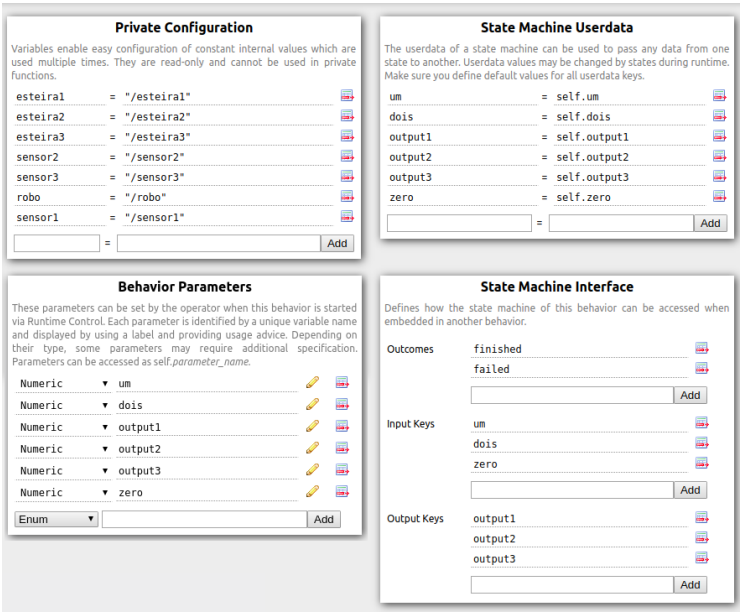
Fonte: (**Autor**)

5.4 Implementação no FlexBE

Para implementar os supervisórios calculados no FlexBE, deve-se primeiro declarar as variáveis que irão armazenar as mensagens enviadas e recebidas e os tópicos utilizados para comunicação. Isto é obtido através do "Behavior Dashboard".

Na comunicação entre FlexBE e o simulador V-REP, serão usados os tópicos "/esteira1", "/esteira2" e "/esteira3" para acionar as esteiras. Para receber as informações dos sensores, serão utilizados os tópicos "/sensor1", "/sensor2" e "/sensor3" numerados de acordo com sua respectiva esteira. O tópico "/robo" também será criado, com a finalidade comandar o robô manipulador e também servirá para informar qual esteira deverá ser ligada. Na Figura 17 pode ser vista a configuração do "Behavior Dashboard".

Figura 17: Configuração do "Behavior Dashboard" no FlexBE.



Fonte: (**Autor**)

Com o intuito de possibilitar a geração de um gráfico monitorando os tópicos e poder visualizar a ocorrência dos eventos ao longo do tempo, foi escolhido que nos tópicos de comunicação serão publicadas mensagens numéricas do tipo "std_msgs/Int8". Estas poderão ter seu valor 0, 1 ou 2. Na Tabela 2 pode ser visto o significado de cada mensagem de acordo com o tópico.

Tabela 2: Significado das mensagens publicadas nos tópicos.

Tópico	Mensagem	Significado
/esteira1	0	Desliga esteira central
	1	Liga esteira central
/esteira2	0	Desliga esteira lateral da direita
	1	Liga esteira lateral da direita
/esteira3	0	Desliga esteira lateral da esquerda
	1	Liga esteira lateral da esquerda
/sensor1	0	Nenhuma caixa detectada na esteira central
	1	Caixa pequena detectada na esteira central
	2	Caixa grande detectada na esteira central
/sensor2	0	Nenhuma caixa detectada na esteira lateral da direita
	1	Caixa detectada na esteira lateral da direita
/sensor3	0	Nenhuma caixa detectada na esteira lateral da esquerda
	1	Caixa detectada na esteira lateral da esquerda
/robo	0	Comanda robô manipulador para a posição central
	1	Comanda ele para pegar a caixa e levar na esteira da direita
	2	Comanda ele para pegar a caixa e levar na esteira da esquerda

Fonte: Autor.

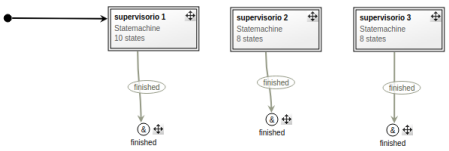
Na tela “Statemachine Editor” será dado início à construção do controle que implementará os supervisórios calculados no DESTool. O primeiro passo para isso é adicionar um “Concurrency Container”, que irá possibilitar criar dentro dele as três máquinas de estado e executá-las ao mesmo tempo. Com isso feito, dentro dele será adicionado três “Statemachine Cointainers”, onde de fato será modelado o comportamento dos supervisórios.

É importante garantir que as variáveis instanciadas em “Input Keys” e “Output Keys” no “Behavior Dashboard” sejam passadas para as respectivas máquinas de estado. Desta forma, serão usadas as funções que foram introduzidas no exemplo do capítulo “Metodologia”. Para os eventos controláveis, o qual se comanda uma das esteiras ou então o robô manipulador, será usada a função “PublisherIntState” informando a devida mensagem numérica e o tópico que ela será enviada.

Para os eventos não controláveis, que serão os sinais dos sensores, ou então para identificação do estado do robô, se ele deve levar a caixa para a esteira da direita ou esquerda, será usada a função “SubscriberState” seguido da função “DecisionState”. A primeira irá fazer a leitura da última mensagem publicada no respectivo tópico e irá armazená-la na variável “output”, enquanto a segunda irá comparar esta com um valor pré determinado, como 0 ou 1, na função “lambda X”.

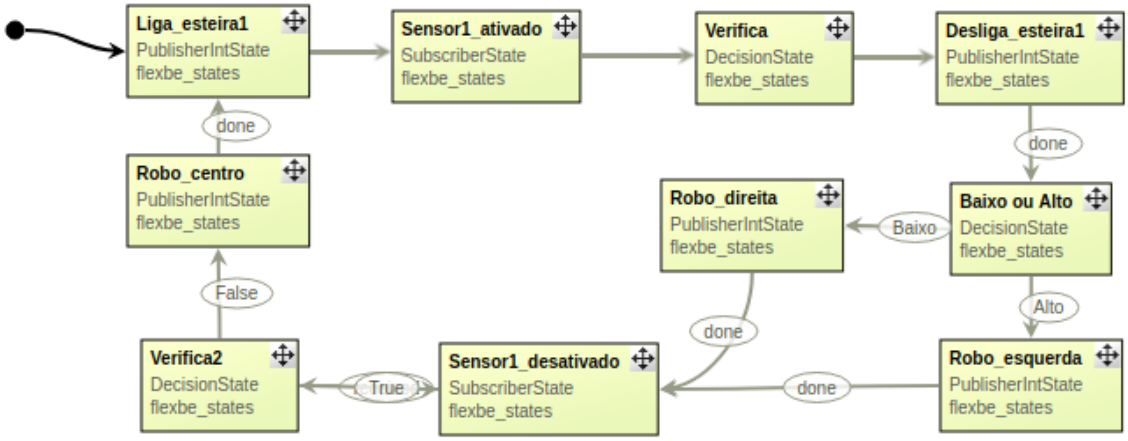
A estrutura montada no FlexBE, bem como os supervisórios 1, 2 e 3, podem ser vistos, respectivamente, nas figuras 18, 19, 20 e 21.

Figura 18: Supervisórios - FlexBE.



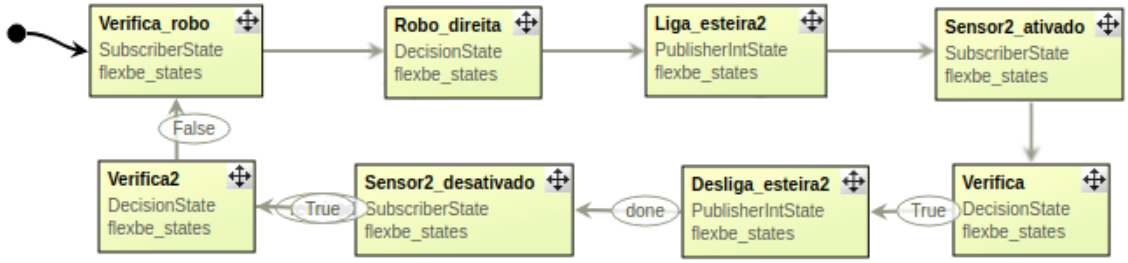
Fonte: (**Autor**)

Figura 19: Supervisório 1 - FlexBE.



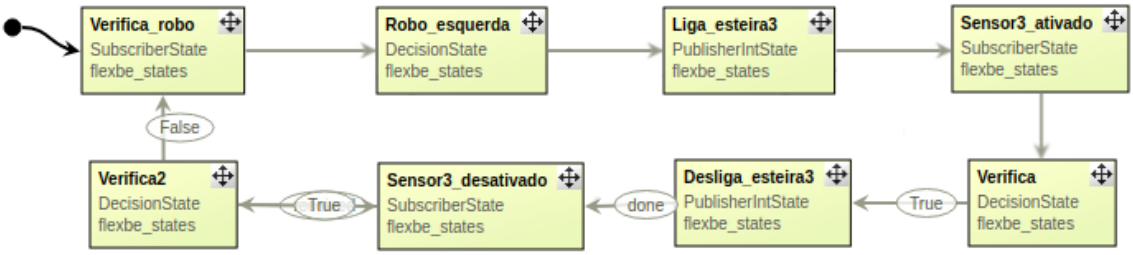
Fonte: (**Autor**)

Figura 20: Supervisório 2 - FlexBE.



Fonte: (**Autor**)

Figura 21: Supervisório 3 - FlexBE.



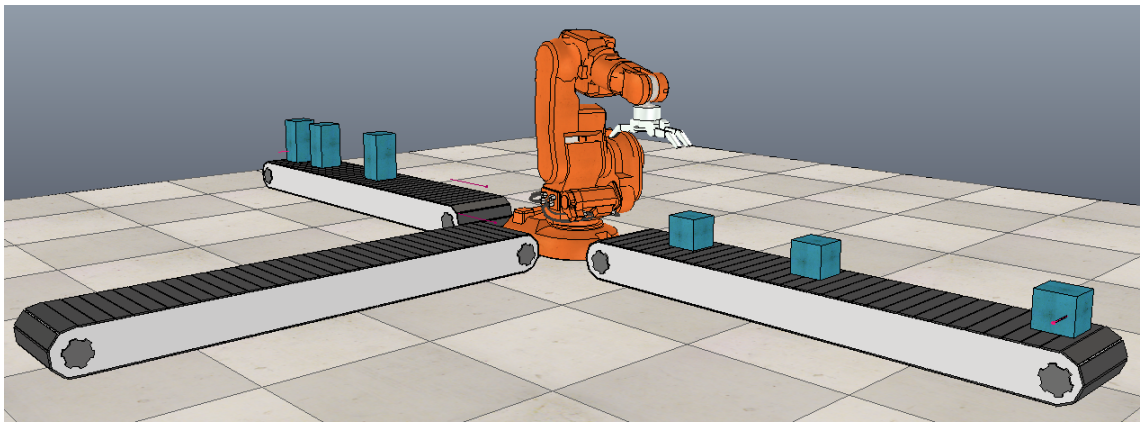
Fonte: (**Autor**)

5.5 Comunicação com o V-REP

Para verificar o funcionamento e testar os supervisórios foi feito, na planta montada no V-REP, uma interface de comunicação com o ROS, da mesma forma como foi abordado no capítulo "Metodologia". Em cada conjunto de esteira + sensor, foi criado um *publisher* para publicar o estado do sensor no seu respectivo tópico e um *subscriber* para receber as mensagens que comandam o funcionamento das esteiras. E por fim, no robô manipulador também foi criado um *subscriber* que é responsável por ler as mensagens que informam se o robô deve levar a caixa para a esteira da direita ou para a da esquerda. Ao realizar a

simulação, o comportamento projetado na síntese dos controladores foi obtido, onde foi detectado corretamente o tamanho das caixas e informado ao robô de modo que ele as separou como desejado. O resultado da simulação pode ser visto na Figura 22.

Figura 22: Final da simulação - V-REP.

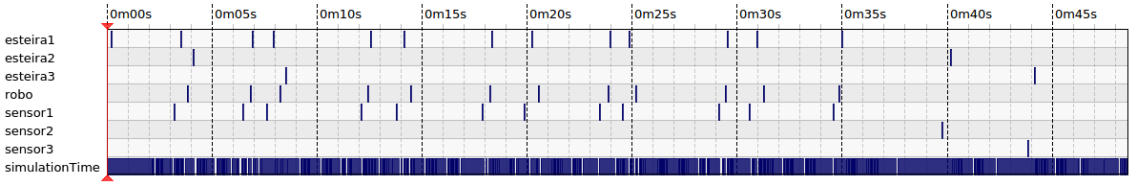


Fonte: (**Autor**)

Entretanto, ao longo da montagem do cenário da linha automatizada e dos primeiros testes, diversos aspectos construtivos e de execução puderam ser repensados ao serem encontrados erros no funcionamento do sistema. O primeiro deles foi o posicionamento dos dois sensores de presença na esteira central e como fazer para que fosse enviada a correta mensagem com a altura da caixa ao supervisório, que causou alguma dificuldade. Em seguida, foi observada a necessidade de alterar a restrição modelada entre a esteira central e o robô, visto que as caixas estavam chegando enquanto o robô ainda estava transportando outras caixas. Por fim, foi possível notar, através dos gráficos, que o robô passava em frente ao sensor de presença, gerando comportamentos estranhos que então, puderam ser contornados. Tais situações reforçam a importância de simular o sistema antes da instalação na fábrica, esta que foi uma das principais motivações.

Como comentado anteriormente, foi pensado na utilização de mensagens numéricas com o intuito de poder gerar gráficos dos sinais ao longo do tempo, para uma análise mais detalhada do funcionamento dos supervisórios. E visto que cada simulação é diferente da outra, julgou-se importante primeiramente fazer a gravação dos sinais para que a simulação não precisasse ser executada mais de uma vez para análise de diferentes sinais. Para tal, foi executada a funcionalidade “rqt_bag” do ROS, que é usada e recomendada para monitorar tópicos especificados e gravar todas as mensagens que são publicadas neles. Desta forma, foi configurado para que ele monitorasse todos os tópicos utilizados para a comunicação, assim como o tópico que é utilizado pelo V-REP para publicação do tempo de simulação. Os sinais gravados por ele podem ser vistos na Figura 23.

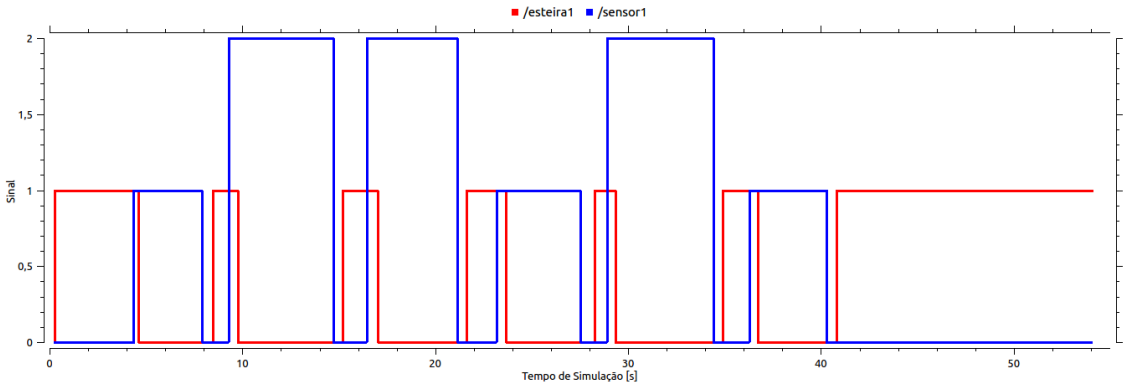
Figura 23: Sinais gravados no “rqt_bag”.



Fonte: (**Autor**)

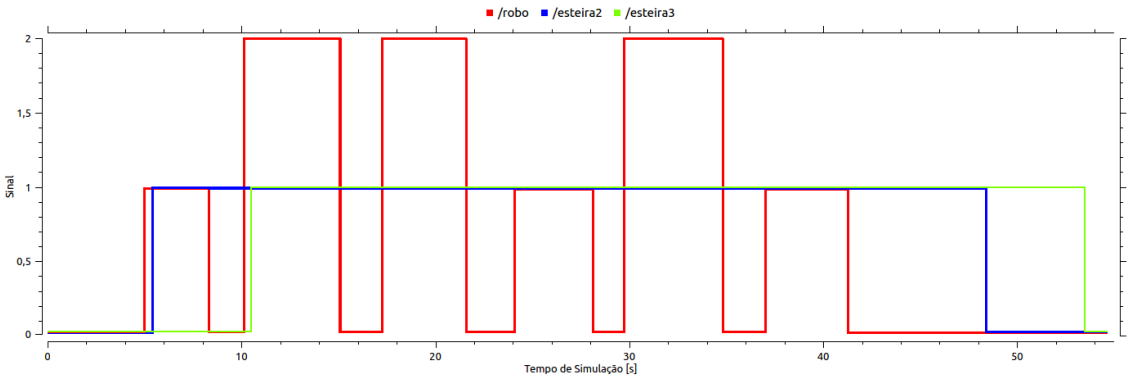
Com estes sinais gravados, é então, possível executá-los quantas vezes forem necessárias para geração e análise dos gráficos. Onde o próprio "rqt_bag" é capaz de publicar as mensagens salvas nos tópicos correspondentes. Assim, utilizando a ferramenta "rqt_multiplot" também do ROS, serão então, gerados os gráficos dos sinais do sistema que ocorreram durante a simulação. Para melhor compreensão, os sinais foram separados em dois gráficos. São eles os eventos da esteira central e do "sensor1", na Figura 24, e os eventos do robô e das esteiras laterais na Figura 25.

Figura 24: Gráfico da simulação - Esteira central e Sensor 1.



Fonte: (**Autor**)

Figura 25: Gráfico da simulação - Robô e esteiras laterais.



Fonte: (**Autor**)

No primeiro gráfico, Figura 24, observa-se a ocorrência de seis subidas do sinal do sensor da esteira central, em azul. Onde três deles são para o valor um, correspondente às caixas pequenas, e três deles para o valor dois, correspondente às caixas grandes. Sempre após estes sinais, ocorre o desligamento da esteira central, sinal vermelho, quando as caixas chegaram em sua extremidade. O sensor permanece ativado enquanto o robô está se descolando para pegar a caixa e retirá-la dali. Quando isto ocorre, o sensor tem seu valor retornado a zero, e então a esteira volta a ser ligada.

É importante observar que o tempo entre a detecção do sensor e o desligamento da esteira não é constante. Isto varia de acordo com o que está sendo processado no computador naquele momento, podendo assim, ter interferência nos tempos de ocorrência dos eventos. Inclusive, é visível que este intervalo na primeira detecção é menor do que nas demais e isto se deve ao fato de o robô ainda estar parado neste momento. Tal fato é

explicado pela dificuldade computacional para o controle de posicionamento de todos os graus de liberdade do robô ao longo de sua movimentação.

O segundo gráfico, Figura 25, complementa os sinais do primeiro. Nele são encontrados também seis subidas no sinal do robô manipulador, em vermelho, representando as seis caixas que ele deve transportar. Quando seu sinal vai para um, significa que é uma caixa pequena, que deve ser levada para a esteira da direita e quando vai para dois, é uma caixa grande, que deve ser levada para a esteira da esquerda. Analisando os dois gráficos é possível observar que a sequência de eventos do robô tem sua ocorrência logo após o "sensor1" detectar as caixas e então desligar a esteira central. Adicionalmente a isto, é visível a que a esteira da direita, em azul, é ligada logo que o robô recebe o valor um e a esteira da esquerda, em verde, logo após ele receber o valor dois. Elas então, permanecem ligadas transportando as caixas até estas chegarem às suas extremidades, sendo desligadas com a detecção pelos seus respectivos sensores.

6 CONCLUSÃO E TRABALHOS FUTUROS

6.1 Conclusão

Mais uma vez, como em muitas outras pesquisas envolvendo sistemas a eventos discretos, a metodologia de autômatos e controle supervisório mostrou-se adequada e recomendável à modelagem deste tipo de sistema, bem como para a síntese de seus controladores. Utilizando-se a teoria de controle modular local, foi modelado e construído, através de pequenos autômatos que representavam o funcionamento de cada equipamento e também da interdependência entre eles, supervisórios modulares que tinham suas ações complementadas e foram capazes de controlar a planta completa. Adicionalmente a isto, é preciso ressaltar a importância da ferramenta DESTool, que esteve por trás de toda esta síntese dos controladores e facilitou todos os cálculos envolvidos.

Ao término deste trabalho, pôde-se perceber que, apesar da necessidade de tradução dos estados e eventos do DESTool, o FlexBE mostrou ser uma ferramenta com uma interface gráfica que de fato facilitava o entendimento do supervisório, para possíveis alterações e manutenções. E, além disso, a implementação nele possibilitou a simulação no *software* proposto e também a utilização em demais simuladores com comunicação através do ROS. Com diversos exemplos de cenários e com a facilidade de programação e comunicação, o *software* V-REP foi de fundamental importância para o cumprimento dos objetivos propostos. Com sua utilização, rapidamente foi feita uma interface de comunicação com os supervisórios através do ROS e assim, pôde-se verificar o correto funcionamento do sistema automatizado.

É necessário também citar a importância do ROS no desenvolvimento deste trabalho. Mesmo sem possuir um *software* propriamente dito, que fosse possível o ver, ele esteve por trás de todo o funcionamento da implementação dos supervisórios, desde a construção da máquina de estados no FlexBE, até a verificação do funcionamento dos supervisórios no V-REP. Com diversas ferramentas de gravação das mensagens dos tópicos, de geração de gráficos e até com a possibilidade de rapidamente ver o que estava sendo publicado em cada tópico através de simples comandos no terminal do sistema operacional, ele se mostrou completo e possibilitou a garantia de que os softwares estavam se comunicando e enviando as devidas mensagens.

Visto isso, pode-se considerar que os objetivos desejados no início do trabalho foram cumpridos. Com uma metodologia de baixo custo, utilizando apenas *softwares* livres, ROS e FlexBE, ou então com versão educacional, V-REP, foi encontrada uma nova forma de implementação de controle supervisório a Sistemas a Eventos Discretos. Esta que se mostrou com grande potencial de utilização, permitindo fácil simulação e manutenção, que ajudarão no projeto e experimentação de novas linhas de produção, fato o qual motivou este trabalho.

6.2 Trabalhos Futuros

Em futuros trabalhos, sugere-se a aplicação da metodologia estudada neste trabalho em um sistema real, utilizando pesquisas já realizadas sobre a comunicação entre ROS e CLPs. Nestas pesquisas, é utilizada uma interface de comunicação onde constantemente é solicitado ao CLP os valores de determinados registradores, que contém dados dos sensores, e quando um destes é alterado, este é disponibilizado no respectivo tópico criado para comunicação com o supervisório. Enquanto isso, há funções que supervisionam determinados tópicos e sempre que o supervisório envia alguma mensagem, está é direcionada ao respectivo registrador do CLP que então aciona ou para algum equipamento.

Com tal aplicação, é possível realizar uma análise mais aprofundada em questões como tempo de processamento do supervisório, atrasos de comunicação e também confirmar se com um processador dedicado a um robô industrial os tempos de resposta são mais rápidos entre detecção do sensor e acionamento de determinado equipamento, como havia sido comentado nos resultados da simulação.

Visto que para sistemas mais complexos é esperada a necessidade de mais do que três supervisórios para controlar a planta completa com a metodologia estudada no presente trabalho, a tarefa de traduzir os estados e eventos do DESTool para o FlexBE de forma manual pode se tornar complicada. Desta forma, é indicado para um trabalho futuro desenvolver uma ferramenta que automatizasse este processo. Utilizando um padrão de nomenclatura para os estados e eventos, onde estes contivessem a mensagem e o tópico para o qual elas devessem ser enviadas, poderia ser encontrada uma forma que facilitasse esta transição e a implementação dos supervisórios.

Recomenda-se também testar demais funcionalidades permitidas pelo ROS, visto que sua comunidade de usuários está em constante desenvolvimento de ferramentas e integração com novos softwares. Neste sentido, podem ser utilizadas bibliotecas de visão computacional que poderiam fazer a automática identificação de objetos. Neste trabalho foi feita uma abordagem simples de identificação das caixas, utilizando apenas dois sensores de presença em alturas diferentes, entretanto poderia ter sido feito com captura e processamento de imagem, que então identificaria os diferentes tipos de caixa. Estudos destas abordagens estão cada vez mais sendo realizados, visto a crescente utilização de sistemas de automática identificação de objetos como em sistemas de manufatura flexíveis.

Aquisição de dados e gerenciamento de produção também poderiam ser realizados juntamente com a metodologia estudada. Utilizando a facilidade de obtenção das mensagens enviadas nos tópicos do ROS, seria possível armazenar todos os dados necessários da linha de produção em um banco de dados ou então integrá-los com ferramentas de gestão que permitem geração de gráficos e análise dos dados obtidos para gerenciamento da produção.

BIBLIOGRAFIA

BANKS, J. **Discrete-Event System Simulation**. [S.l.]: Pearson Prentice Hall, 2005. ISBN 978-0-13-815037-2.

CASSANDRAS, C. G.; LAFORTUNE, S. **Introduction to Discrete Event Systems**. Berlin, Heidelberg: Springer-Verlag, 2006. ISBN 978-0-387-33332-8.

CURY, J. E. R. **Teoria de Controle Supervisório de Sistemas a Eventos Discretos**. Mini Curso V Simpósio Brasileiro de Automação Inteligente, Canela-RS: [s.n.], 2001. Disponível em: <https://docplayer.com.br/12593760-Teoria-de-controle-supervisorio-de-sistemas-a-eventos-discretos.html>. Acesso em: 10 jun. 2019.

GROOVER, M. P. **Automação Industrial e Sistemas de Manufatura**. São Paulo, SP: Pearson Prentice Hall, 2011. ISBN 978-85-4301-501-9.

IEEE Std 100-2000: The Authoritative Dictionary of IEEE Standards Terms, Seventh Edition. [S.l.]: IEEE, 2000. Disponível em: <https://books.google.com.br/books?id=5CV8AQAACAAJ>. Acesso em: 10 mai. 2019.

MOOR, T. **DesTool**. [S.l.: s.n.], 2019. Disponível em: ¡https://www.rt.tf.fau.de/FGdes/destool/¿. Acesso em:

MOOR, T.; SCHMIDT, K.; PERK, S. libFAUDES - An open source C++ library for discrete event systems. In: 2008 9th International Workshop on Discrete Event Systems. [S.l.]: IEEE, 2008. p. 125–130. ISBN 978-1-4244-2592-1. DOI: 10.1109/WODES.2008.4605933.

O'KANE, J. M. **A Gentle Introduction to ROS**. Columbia, SC: CreateSpace Independent Publishing Platform, 2013. ISBN 978-14-92143-23-9.

QUEIROZ, M. H.; CURY, J. E. R. Modular Control Of Composed Systems. In: PROCEEDINGS of the 2000 American Control Conference. Chicago, USA: [s.n.], 2000. p. 4051–4055.

QUEIROZ, M. H.; CURY, J. E. R. Synthesis and Implementation of Local Modular Supervisory Control for a Manufacturing Cell. In:

ROHMER, E.; SINGH, S. P. N.; FREESE, M. V-REP: a Versatile and Scalable Robot Simulation Framework. In: PROCEEDINGS of The International Conference on Intelligent Robots and Systems (IROS). Tokyo, Japan: [s.n.], 2013.

SCHILLINGER, P.; KOHLBRENCHER, S.; VON STRYK, O. Human-Robot Collaborative High-Level Control with an Application to Rescue Robotics. In: IEEE International Conference on Robotics and Automation. Estocolmo, Suécia: [s.n.], 2016.

VIEIRA, A. D. et al. A Method for PLC Implementation of Supervisory Control of Discrete Event Systems. In: IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, VOL. 25, NO. 1. Melbourne, Australia: [s.n.], 2017.