Recebido: 03/02/2016
Aprovado: 01/09/2016

Comissionamento de uma linha de produção no software de simulação a eventos discretos Ururau

Commissioning of a production line in the discrete event simulation software Ururau

Quézia Manuela Gonçalves Laurindo[1] - Universidade Cândido Mendes - Campos dos Goytacazes
João José de Assis Rangel[2] - Universidade Cândido Mendes - Campos dos Goytacazes
Túlio Almeida Peixoto[3] - Universidade Cândido Mendes - Campos dos Goytacazes
Eder Reis Tavares[4] - Universidade Cândido Mendes - Campos dos Goytacazes
Fábio Freitas da Silva[5] - Universidade Cândido Mendes - Campos dos Goytacazes
Ítalo de Oliveira Matias[6] - Universidade Cândido Mendes - Campos dos Goytacazes

RESUMO Objetivo deste trabalho foi demonstrar o funcionamento de um modelo de simulação a eventos discretos, construído com o software Ururau integrado a um controlador lógico programável para ser empregado em comissionamento de sistemas de controle. O trabalho apresenta ainda diversos testes realizados para ilustrar diferentes possibilidades de comissionamento com o software. Os resultados mostraram a viabilidade do Ururau em ser empregado com este propósito e, também, o mecanismo interno responsável pelo funcionamento da comunicação de um modelo de simulação e um controlador digital.

Palavras-chave Ururau. Simulação a eventos discretos. Sistemas de controle.

ABSTRACT *The aim of this work was to demonstrate the utilization of a discrete event simulation model built with the Ururau software integrated to a programmable logic controller to be applied in commissioning of control systems. The paper also presents different tests to illustrate various possibilities of commissioning with that software. The results showed the viability of the Ururau to be employed with this purpose and the internal mechanism responsible for the communication between a simulation model and a digital controller.*

Keywords *Ururau. Discrete event simulation. Control systems.*

1. Rua Ibitiúva, 151, Padre Miguel, Rio de Janeiro, RJ, CEP: 21715-400, manuelaglaurindo@gmail.com
2. joao.rangel@ucam-campos.br
3. tulioap@gmail.com
4. ederreis-@hotmail.com
5. fabio1_freitas@hotmail.com
6. italo@ucam-campos.br

LAURINDO, Q. M. G.; RANGEL, J. J. A.; PEIXOTO, T. A.; TAVARES, E. R.; SILVA, F. F.; MATIAS, I. O. Comissionamento de uma linha de produção no software de simulação a eventos discretos Ururau. **GEPROS. Gestão da Produção, Operações e Sistemas**, Bauru, Ano 12, nº 1, jan-mar/2017, p. 167-191.

DOI: 10.15675/gepros.v12i1.1614

1. INTRODUÇÃO

O comissionamento de plantas industriais é uma das etapas realizada nos projetos de engenharia que visa garantir o pleno funcionamento de equipamentos, máquinas e sistemas de controle. Normalmente, o comissionamento é realizado através de um conjunto de procedimentos e testes que tornam a unidade industrial apta ao pleno funcionamento. Seu objetivo principal é assegurar, de forma coordenada e segura, a transferência da unidade industrial do projetista para o operador. Em muitos casos, o comissionamento pode ser realizado com auxílio de simulação computacional e diversos *softwares* de simulação possuem integração com equipamentos de sistemas de controle para a realização deste propósito (VERSTEEGT; VERBRAECK, 2002).

A possibilidade de integração de modelos de simulação com sistemas de controle foi demonstrada há quase duas décadas em trabalhos como os de Dougall (1998) e Banks (2000). Estes autores levantaram ainda a questão sobre a possibilidade de os *softwares* de simulação a eventos discretos (SED) também poderem ser utilizados nos comissionamentos de sistemas de controle. Assim, os *softwares* de SED além de poderem ser empregados na análise de uma determinada unidade industrial com a finalidade de se avaliar a utilização de um recurso, por exemplo, podem também ser usado na etapa de comissionamento. Ou seja, uma vez construído um modelo de SED para se avaliar um determinado parâmetro de um sistema, este mesmo modelo pode ser utilizado, posteriormente, para testar o sistema de controle e, assim, poder auxiliar, também, no comissionamento da respectiva unidade.

Assim, de uma forma geral, pode-se dizer que a simulação permite desenvolver e testar sistemas automatizados de uma maneira segura, com menor tempo e custos, além de possibilitar a análise dos mesmos. Segundo Jain et al. (2012) a SED é mais apropriada para modelagem de processos de sistemas e para avaliação de questões relacionadas a recursos. Este é um método que permite projetar e avaliar novos sistemas, efetuar reconfigurações de leiaute e mudanças nas regras de controle (SAKURADA; MIYAKE, 2009). Existem, mundialmente, mais de cinquenta *softwares* comerciais para o desenvolvimento de modelos de SED(SWAIN, 2013). Atualmente, um grande número desses *softwares* está em uso no Brasil e possuem capacidade de se comunicar com sistemas de controle e podem auxiliar em comissionamentos (RANGEL et al., 2012).

Recentemente, foi apresentado por Peixoto, Rangel e Matias (2015) o primeiro *software* de SED desenvolvido no Brasil, sendo livre de custos e de código fonte aberto. Os autores apresentaram uma descrição detalhada do respectivo *software*, denominado de Ururau e a capacidade dos seus modelos poderem ser executados com comunicação com controladores digitais. Pesquisadores, como King e Harrison (2010), destacam que *softwares* livres são mais fáceis de serem customizados por permitirem acesso ao código fonte.

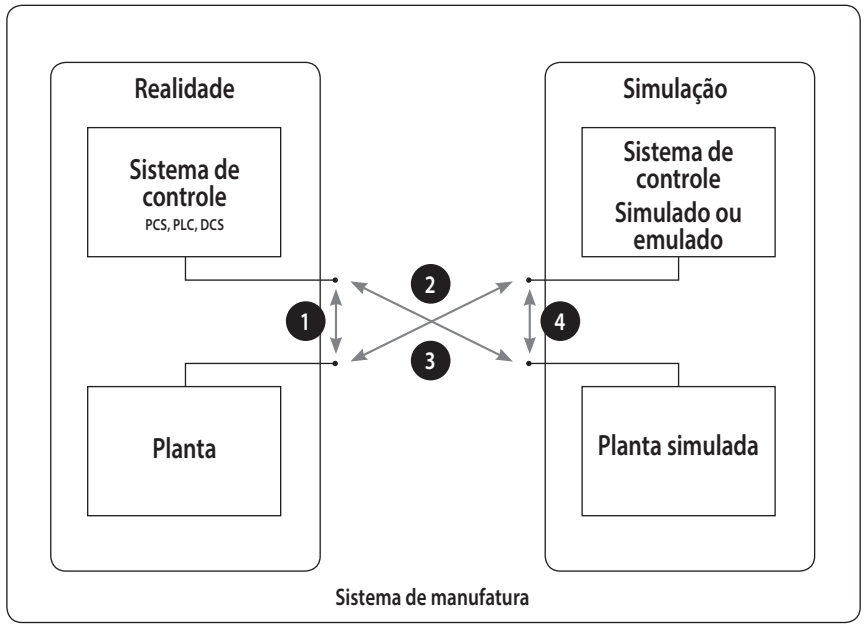
Portanto, este trabalho tem objetivo de demonstrar a possibilidade de utilização deum *software* de SED gratuito como uma ferramenta viável e de baixo custo de utilização para auxiliar em comissionamentos de plantas industriais. Assim, é descrito de forma detalhada o mecanismo de funcionamento de um modelo de SED construído com o *software* Ururau integrado a um controlador lógico programável (CLP) para ser empregado em comissionamento de um sistema de controle. Esta é uma ferramenta de simulação recomendada, principalmente, a estudantes e empresas de pequeno porte. Além disso, foram demonstrados neste trabalho diversos procedimentos que podem ser empregadas durante um comissionamento.

2. REFERENCIAL TEÓRICO

2.1. Comissionamento com auxílio de simulação

Ao se adotar *softwares* de simulação em comissionamentos, um CLP, por exemplo, pode ficar interligado a um modelo de simulação no lugar do sistema real. Isso garante um ambiente mais seguro e menos dispendioso no momento do teste. Desta forma, os sistemas de controle podem ser testados de forma mais eficaz e flexível (SMITH; CHO, 2008). A Figura 1 mostra as formas mais usuais pelas quais um comissionamento pode ser executado.

Figura 1 – Possibilidades de realizar o comissionamento.



Fonte: Auinger et al. (1999).

Ainda na Figura 1, a seta 1 indica o comissionamento tradicional, que é realizado entre uma planta real e um controlador real. A seta 2 representa o comissionamento virtual feito utilizando uma planta simulada e um controlador real. A seta 3 utiliza um controle simulado e uma planta real representando o comissionamento *reality in the loop*, que possibilita ao modelo adquirir maior realismo. A seta 4 representa o comissionamento *offline*, ou seja, a integração entre um controle simulado e uma planta simulada, o que torna possível utilizar a simulação para auxiliar a etapa de desenvolvimento do sistema de controle (AUINGER et al., 1999).

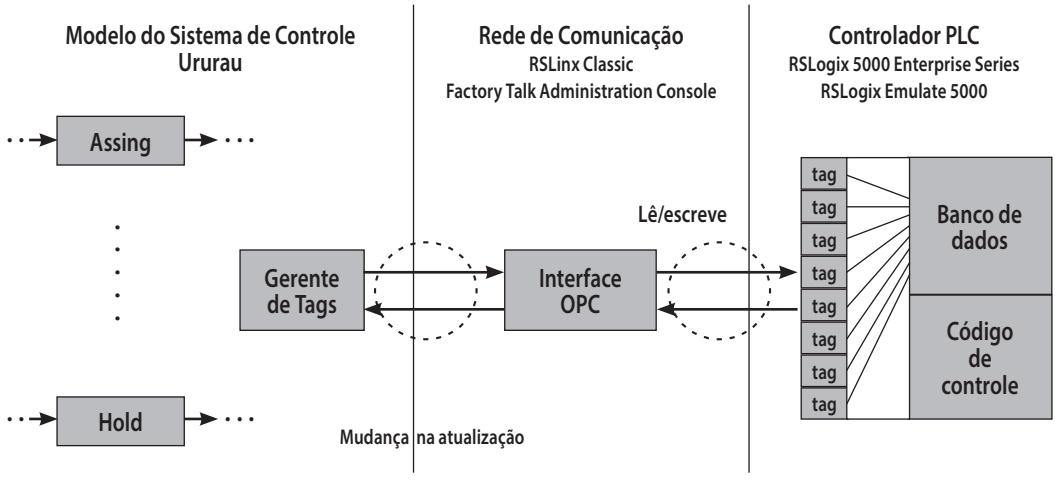
Diferentes trabalhos mostram a possibilidade de utilização de simulação no comissionamento de sistemas de controle. Ko e Park (2014), por exemplo, apresentaram uma metodologia de modelagem para construção de uma planta virtual em conjunto com CLP que pode ser usada para comissionamento. Segundo os autores o objetivo final do comissionamento virtual é proporcionar uma planta virtual para validar sua lógica de controle. Como resultado pode-se alcançar a transição perfeita do ambiente virtual para o real. Já, Carlsson et al. (2012) ressaltam que a verificação de código de simulações com CLPs é parte do comissionamento virtual. A interface *Object Linkingand Embedding for Process Control* (OPC), facilita a conexão entre CLP e ferramentas de simulação e pode ser usada para efeitos de verificação. No entanto, os autores, destacam alguns problemas desta abordagem que podem levar a um resultado de verificação não confiável.

Assim foram identificam quatro grandes problemas com interface OPC e foram apresentadas duas soluções possíveis para os mesmos. Dominka e Bender (2007) afirmam que bugs no *software* controlador podem persistir por causa de discrepâncias inevitáveis entre a realidade e a simulação. Estas falhas podem causar estados críticos da máquina na fase de comissionamento. Então, para realizar o comissionamento de plantas de forma mais segura, sistemática e menos onerosa os autores propuseram uma nova abordagem. Essa abordagem, chamada de comissionamento híbrido, começa por uma simulação do tipo *hardware-in-the-loop* da planta. Gradualmente, a planta simulada é então substituída por partes reais da unidade de produção. Este processo chega ao fim quando a planta, em sua totalidade, for devidamente comissionada.

2.2. Integração de modelos de simulação e sistemas de controle

A integração do modelo de simulação com o sistema de controle pode ser feita através de uma interface OPC, que possibilita o desenvolvimento de *softwares* que acessam dados de uma vasta gama de dispositivos sem qualquer código específico de fabricante (SMITH; CHO, 2008). A Figura 2 ilustra o mecanismo para a integração entre o *software* Ururau e um CLP, através da interface de comunicação OPC.

Figura 2 – Integração do modelo de simulação Ururau com o sistema de controle.



Fonte: Adaptado de Smith e Cho (2008).

Existem dois módulos de processos no Ururau que são utilizados para que haja a integração entre o modelo de simulação e o sistema de controle: o *Assign* e o *Hold*. O *Assign* é usado para gerar o acionamento de uma entrada para o sistema de controle, já o *Hold* é utilizado no envio do sinal de saída do sistema de controle para o Ururau. Desta forma, é possível estabelecer conexões entre as variáveis do modelo de simulação com os *tags* da lógica de controle do CLP, por meio da ligação gerada entre o ambiente de simulação e o servidor de dados OPC. Os *tags* estabelecidos no *software* de programação do CLP referem-se às entradas e saídas localizadas em sua memória. As variáveis definidas no Servidor OPC são utilizadas dentro desses módulos. No entanto, só é possível ter acesso a esses dados quando a comunicação OPC estiver habilitada.

A comunicação entre o modelo de simulação e o sistema de controle é *duplex*, pois o modelo de simulação escreve dados na interface OPC e também faz a leitura de dados de comunicador. Assim, o sistema de controle lê dados do Servidor OPC e faz a escrita de informações no comunicador. As variáveis utilizadas para leitura e escrita são do tipo booleanas. Essas assumem, apenas, valores de nível lógico 0, para a condição falsa, e nível lógico 1, para a condição verdadeira.

3. MÉTODO

Segundo Banks (2010), um problema que pode ser analisado por meio de modelagem deve ser bem formulado, com objetivos e delimitações bem definidos. Para isso, um esboço gráfico deve servir de auxílio como, por exemplo, um modelo conceitual. Os dados das etapas devem ser bem fundamentados para que o modelo corresponda, o mais próximo possível, ao sistema real. Por fim, o modelo deve ser construído numa linguagem apropriada para sua simulação.

Assim, a metodologia proposta por Banks et al. (2010) foi utilizada para o desenvolvimento dos modelos de simulação. Esta metodologia propõe: formulação e análise do problema; construção do modelo conceitual; construção do modelo de simulação; verificação e validação; experimentação e interpretação; e análise estatística dos resultados. Ressalta-se que durante as fases de verificação e validação dos modelos de simulação foram observadas as etapas sugeridas por Sargent (2013).

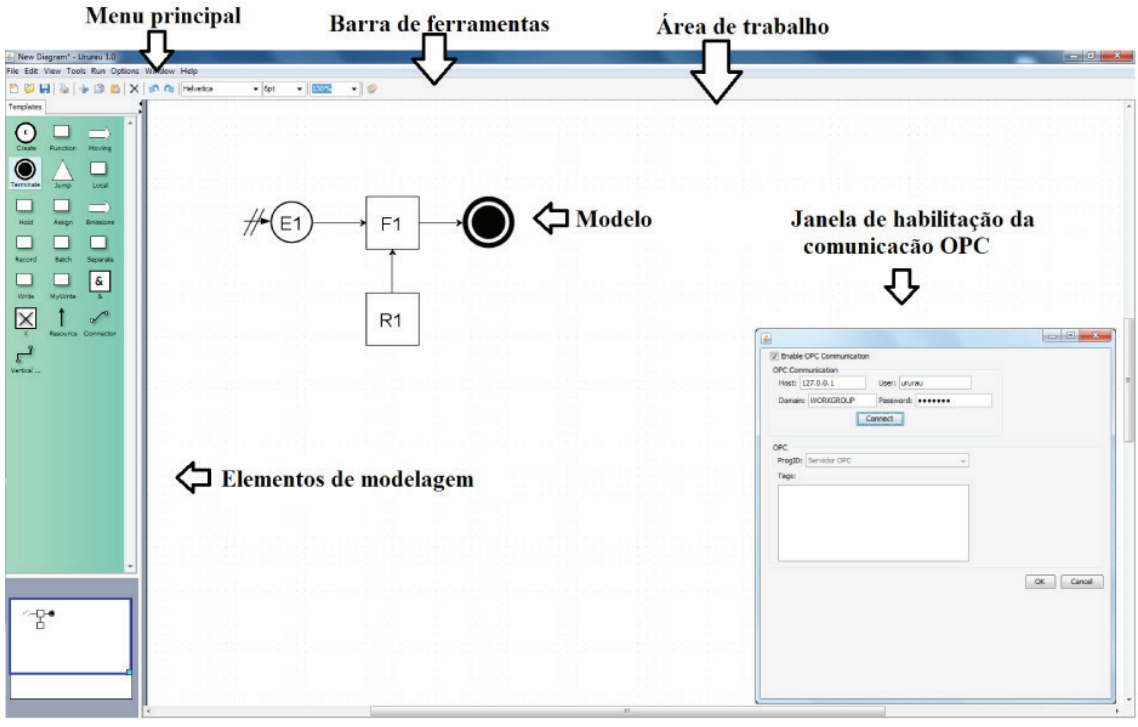
Para a construção do modelo de simulação, um modelo conceitual foi desenvolvido em linguagem IDEF-SIM proposta por Montevechi et al. (2010), esta linguagem permiti uma melhor visualização e compreensão das etapas de produção e suas características.

3.1. *Software* Ururau

Em um recente levantamento de literatura de *softwares* livres para pesquisa operacional, Dagkakis e Heavey (2015), citam o Ururau, em uma relação de quarenta e quatro outros. Os autores ressaltaram como fatos positivos do referido *software*: vídeos tutorias disponíveis em meio eletrônico; comparação dos resultados com ferramenta comercial equivalente; e interface gráfica simples de operar. Este *software* já foi aplicado com CLP no trabalho de Cardoso et al. (2012) e Cardoso et al. (2013). Estes autores apresentam um ambiente hibrido para testes e treinamento em sistemas de controle na fabricação. O Ururau foi integrado a um CLP e às estações didáticas de processos de fabricação utilizados na formação de estudantes.

Assim, o modelo computacional foi implementado no *software* de simulação Ururau. Este *software* tem como base a biblioteca JSL – *Java Simulation Library* proposta por Rosseti (2008). O *software* Ururau possui um ambiente de modelagem simples, em que seus módulos são baseados na linguagem conceitual IDEF-SIM, apresentada por Montevechi et al. (2010). A Figura 3 apresenta a interface gráfica do *software* Ururau. Em Peixoto, Rangel e Matias (2012) foi demonstrada também a aplicação da biblioteca JSL em uma simulação de Monte Carlo.

Figura 3 – Ambiente de simulação do *software* Ururau.



Fonte: Elaborado pelos autores (2015).

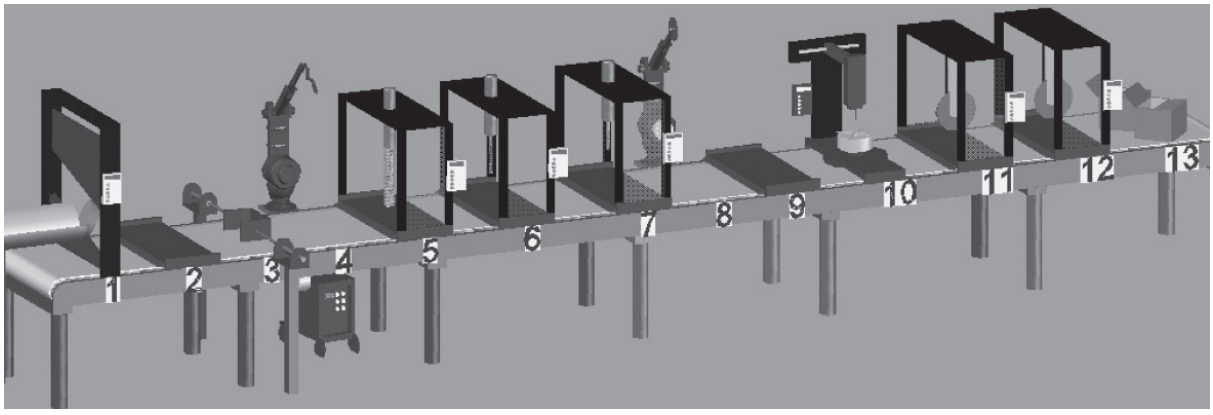
Esta Figura mostra o menu principal, barra de ferramentas, área de trabalho, modelo de simulação, elementos de modelagem e janela de habilitação da comunicação com a interface OPC. Os elementos de modelagem (módulos) representam funções utilizadas para a implementação do modelo de simulação. Para construção dos modelos, basta selecionar os módulos, movê-los até a área de trabalho, estabelecer as ligações entre os elementos de acordo com a lógica do sistema em análise e editar seus parâmetros. Na janela de habilitação OPC quando o usuário clica em "*connect*" e seleciona a opção "Servidor OPC", as variáveis definidas na interface OPC aparecem, no campo "*Tags*", e podem ser usadas no modelo de simulação (PEIXOTO et al., 2015).

Para que ocorra a integração do modelo de simulação com o sistema de controle é preciso definir, no Servidor OPC, as variáveis para serem usadas em módulos no Ururau e no CLP. Também é necessário habilitar no Ururau a comunicação com o Servidor OPC, implementar o modelo de simulação no *software* Ururau e implementar a lógica de controle em um *software* de programação de CLPs. Por fim, transferir a lógica programada para o CLP através de um cabo de rede e executar o modelo de simulação.

3.2. Descrição do sistema

O sistema tratado neste trabalho refere-se a uma linha hipotética de produção de flanges. Todos os processos que constituem a planta industrial podem ser vistos na Figura 4.

Figura 4 – Linha de produção hipotética de flanges: Cortar tarugos (1), Empilhar (2), Alinhar (3), Pontear (4), Furar centro (5), Furar bordas (6), Rosquear (7), Desbastar (8), Desempilhar (9), Tornear (10), Lixar (11), Polir (12) e Embalar (13).



Fonte: Elaborado pelos autores (2015).

A linha de produção de flanges é abastecida com cilindros de 1 metro de comprimento e 30 centímetros de diâmetro, seguindo uma função exponencial de 40 segundos. Esses cilindros são cortados por uma lâmina, no primeiro ponto da linha, em 20 pedaços de 5 centímetros. Após serem cortados, os tarugos formam lotes de 4 peças e são empilhados por um operador. As peças ficam desalinhadas, sendo necessário um alinhamento longitudinal e transversal que são feitos por dois atuadores, acionados após um sensor de proximidade indutiva, montada na esteira, detectar a passagem do lote. Em seguida ao alinhamento, os tarugos empilhados são ponteados com solda em suas laterais para que permaneçam alinhados nos processos subsequentes. Após serem fixados, os tarugos seguem para o processo de furação de centro. A furadeira faz um furo no centro do tarugo com diâmetro de 10 centímetros, que é um terço do seu diâmetro externo.

As peças seguem para o processo de furação das bordas, que é feito em outra furadeira em que uma broca menor está instalada. São feitos 4 furos com diâmetro de 5 centímetros. Após serem feitos os furos nas bordas, estes são rosqueados através do macho. Após isso, o lote de peças é desbastado para que as peças possam ser desempilhadas. A pilha é desfeita pelo mesmo operador que empilhou o lote. Para que fique um ressalto ao redor do furo localizado no centro do tarugo, as peças são torneadas, esse ressalto possui altura e base de 2 centímetros. As peças seguem para o acabamento, são lixadas e depois polidas e para finalizar, um segundo operador embala os flanges.

Por se tratar de um modelo hipotético os tempos dos processos foram arbitrados e os desvios padrões foram fixados em 10%. Foi utilizada a função exponencial na geração de entidades por possuir grande aplicabilidade em sistemas de filas e nos processos optou-se pela utilização da função normal devido à sua importância dentre as distribuições continuas. A Tabela 1 apresenta as informações referentes aos dados do sistema descrito.

Tabela 1 – Descrição dos processos contidos na planta industrial hipotética.

	Descrição	Parâmetros
E2	Entidade: Cilindro	Função: EXPO(40) segundos; 1 por vez; máximo de infinito
F1	Processo: Cortar cilindro	Função: NORM(5,0.5) segundos; quantidade: 1
F2	Processo: Empilhar	Função: NORM(10,1) segundos; quantidade: 1
F3	Processo: Alinhar	Função: NORM(3,0.3) segundos; quantidade: 1
F4	Processo: Pontear	Função: NORM(10,1) segundos; quantidade: 1
F5	Processo: Furar centro	Função: NORM(3,0.3) segundos; quantidade: 1
F6	Processo: Furar bordas	Função: NORM(20,2) segundos; quantidade: 1
F7	Processo: Rosquear	Função: NORM(60,6) segundos; quantidade: 1
F8	Processo: Desbastar	Função: NORM(10,1) segundos; quantidade: 1
F9	Processo: Desfazer pilha	Função: NORM(10,1) segundos; quantidade: 1
F10	Processo: Tornear centro	Função: NORM(7,0.7) segundos; quantidade: 1
F11	Processo: Lixar	Função: NORM(10,1) segundos; quantidade: 1
F12	Processo: Polir	Função: NORM(10,1) segundos; quantidade: 1
F13	Processo: Embalar	Função: NORM(20,2) segundos; quantidade: 1
R1	Recurso: Lâmina	Quantidade: 1
R2	Recurso: Operador	Quantidade: 1
R3	Recurso: Robô	Quantidade: 1
R4	Recurso: Furadeira	Quantidade: 1
R5	Recurso: Furadeira	Quantidade: 1
R6	Recurso: Macho	Quantidade: 1
R7	Recurso: Robô	Quantidade: 1
R8	Recurso: Torno	Quantidade: 1
R9	Recurso: Lixadeira	Quantidade: 1
R10	Recurso: Politriz	Quantidade: 1
R11	Recurso: Operador	Quantidade: 1
L1	Agrupa: Criar lote de 4 peças	Quantidade: 1

Fonte: Elaborado pelos autores (2015).

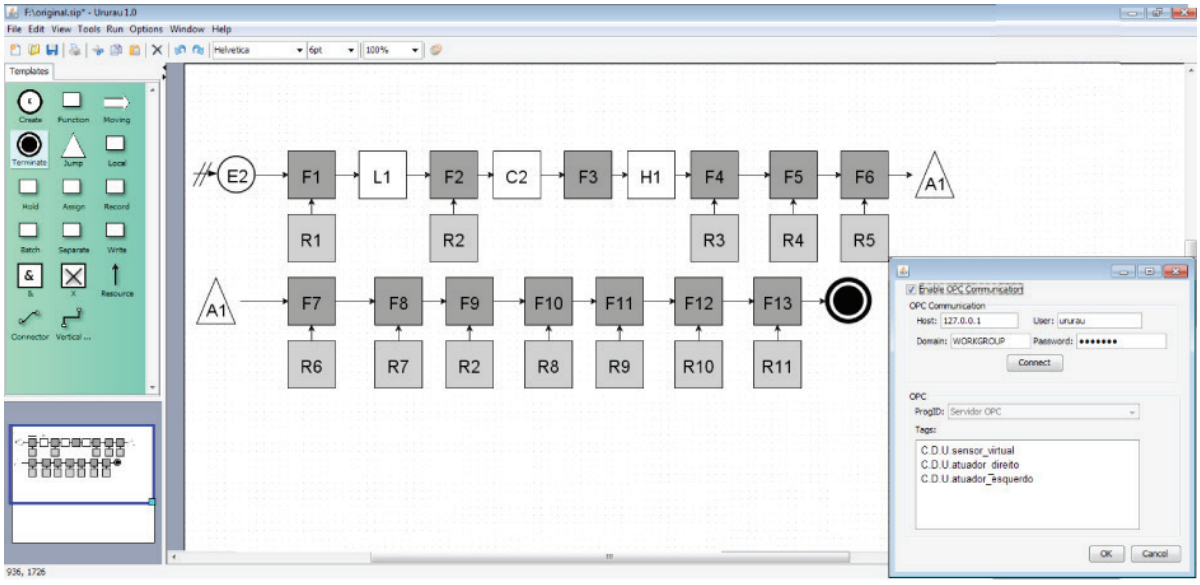
3.3. Modelo de simulação

O modelo de simulação foi construído no *software* livre de custos e de código fonte aberto Ururau. Este modelo é apresentado a seguir na Figura 5.

O módulo (E2) representa a chegada de entidades (cilindros) no sistema. As peças são direcionadas para o primeiro ponto da linha que é o processo de corte dos cilindros (F1), para realizar este processo uma lâmina de corte (R1) é utilizada.

Depois, os tarugos formam lotes de quatro unidades e seguem para o ponto 2 da linha de produção, onde são empilhados (F2) por um operador (R2). As pilhas de tarugos passam pelo processo de alinhamento (F3), para que possam ser furados juntos, de modo que permaneçam alinhados. As pilhas então são ponteadas com solda (F4), através de um robô (R3).

Figura 5 – Modelo de simulação implementado no *software* Ururau.



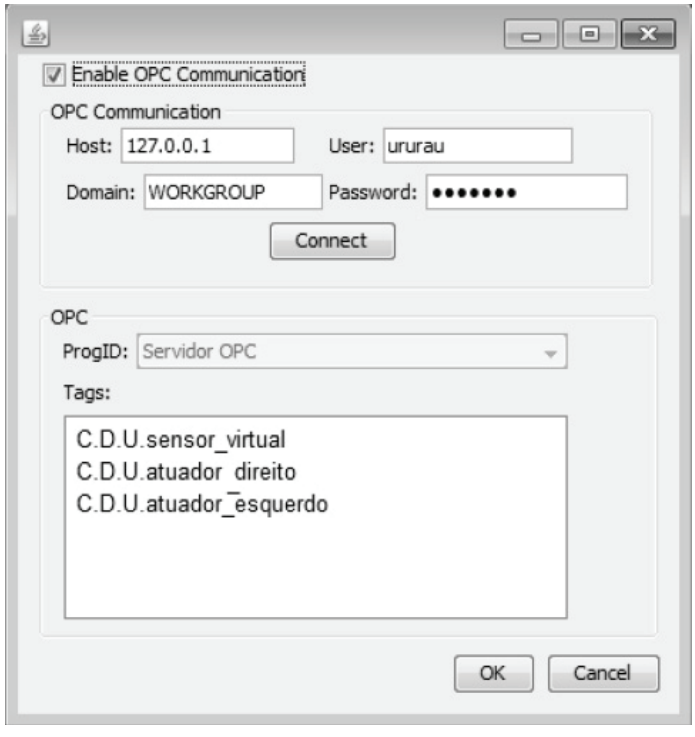
Fonte: Elaborado pelos autores (2015).

Posteriormente, as pilhas passam pela furadeira (R4) no processo de furação de centro (F5). Então, segue para outra furadeira (R5), onde se encontra instalada uma broca com o diâmetro menor, para realizar os furos das bordas (F6).Depois de executados os furos das bordas, estes são rosqueados (F7) utilizando o macho (R6). Os lotes então passam pelo processo de desbaste (F8), por meio de um robô (R7) e em seguida são desempilhados (F9) pelo mesmo operador (R2) que formou as pilhas.

Após serem separados, os tarugos passam pelo processo de torneamento (F10) no torno (R8) que faz um ressalto ao redor do furo no centro dos tarugos. Depois disso, os flanges passam por processos de acabamento, onde são lixados (F11) por uma lixa (R9) e polidos (F12) utilizando uma politriz (R10). Por fim, são embalados (F13) por outro operador (R11). Vale ressaltar que os módulos A1 são utilizados para dar sequência ao modelo, ou seja, não interferem no sistema.

A Figura 6 mostra as variáveis configuradas na interface OPC do Ururau no campo "*Tags*": C.D.U. sensor_virtual, C.D.U. atuador_direito e C.D.U. atuador_esquerdo. A primeira variável citada trata-se de uma variável de entrada. A mesma foi inserida no módulo *Assign* (C2) e, as duas últimas, foram utilizadas no módulo *Hold* (H1) e referem-se à duas variáveis de saída.

Figura 6 – Interface OPC no Ururau.



Fonte: Elaborado pelos autores (2015).

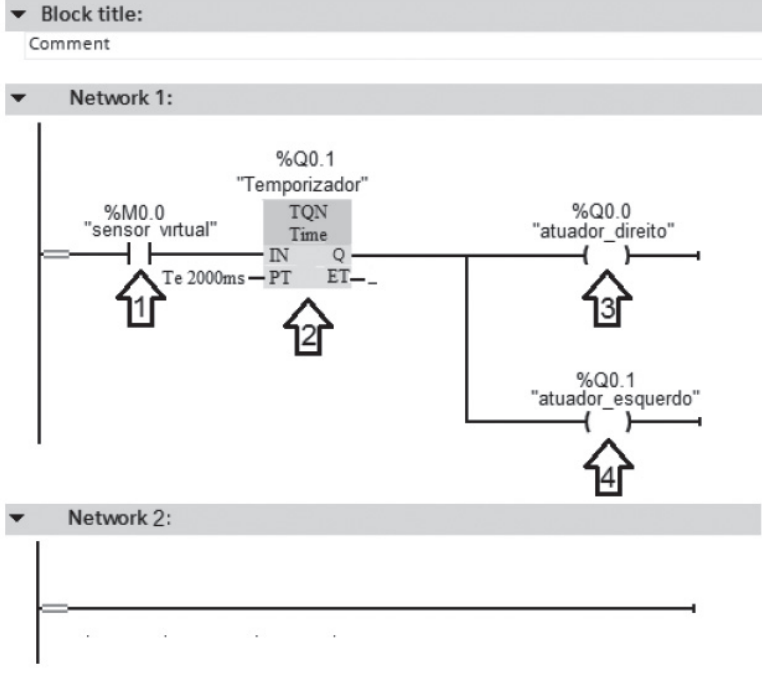
4. RESULTADOS E DISCUSSÃO

4.1. Construção do sistema de controle

A Figura 7 apresenta a lógica de controle implementada no CLP no *software* da SIEMENS. O endereço nomeado M0.0 é virtual, este lê o valor de uma variável, envia um sinal de entrada para o programa em *ladder* no CLP. Assim, o controlador emite um sinal de saída, ou seja, escreve o valor de uma variável acionando as saídas com os endereços (Q0.0 e Q0.1).

Um CLP é composto basicamente por módulos de entrada, módulos de saída e módulos de comunicação. O módulo de entrada (ou contato) pode ler o valor de uma variável e a saída (ou bobina) pode escrever o valor de uma variável. A programação em *ladder*, apresentada na Figura 7, refere-se aos comandos de entradas e saídas que são executados sequencialmente pelo CLP. Esse lê o valor de uma variável de entrada no ponto 1, através do acesso a memória do Ururau. Quando o CLP recebe o sinal de entrada do simulador, este envia o sinal de saída após 2 segundos, devido ao temporizador mostrado no ponto 2. O CLP escreve na memória do Ururau, através das saídas mostradas nos pontos 3 e 4.

Figura 7 – Lógica de controle implementada no CLP. Sensor (1), Temporizador (2), Saída para acionar o avanço do atuador direito (3) e saída para acionar o avanço do atuador esquerdo (4).

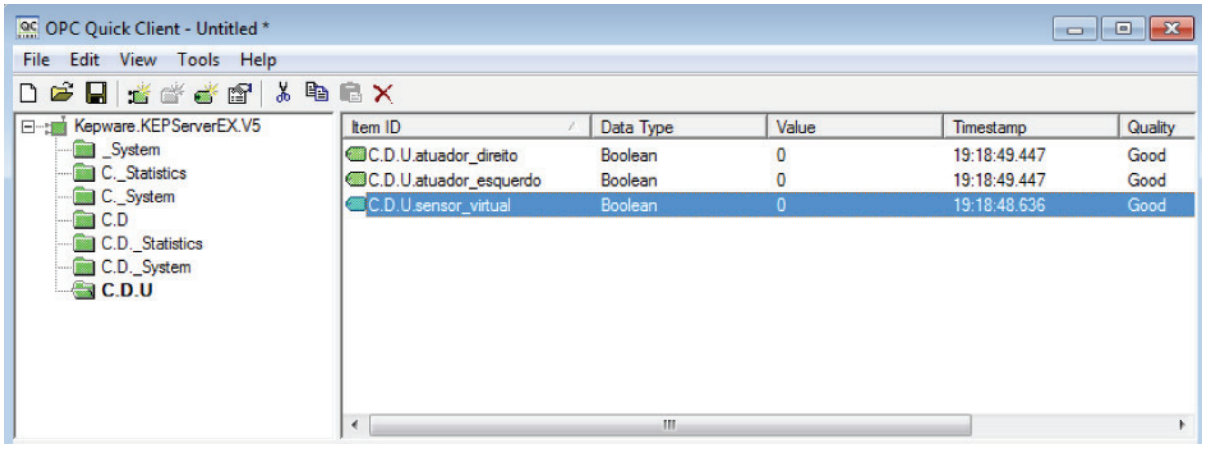


Fonte: Elaborado pelos autores (2015).

7.2. Configuração da interface OPC

A Figura 8 mostra as variáveis configuradas na interface de comunicação do Servidor OPC. Esta Figura apresenta a definição das variáveis C.D.U. sensor_virtual, C.D.U. atuador_direito e C.D.U. atuador_esquerdo. Essas variáveis foram mostradas na Figura 6.

Figura 8 – Interface OPC.



Fonte: Elaborado pelos autores (2015).

7.3. Comissionamento tradicional

Nesta etapa, somente foi abordada a parte física do sistema, ou seja, a bancada pneumática e seus componentes. Um controlador real executou comandos na planta didática. A entrada e as duas saídas foram alimentadas. O simulador Ururau não foi utilizado nesta etapa de teste.

A lógica *ladder* aplicada no comissionamento tradicional foi a mesma utilizada nos outros testes. Essa lógica permanece gravada no CLP até que seja feito o carregamento de outra lógica.

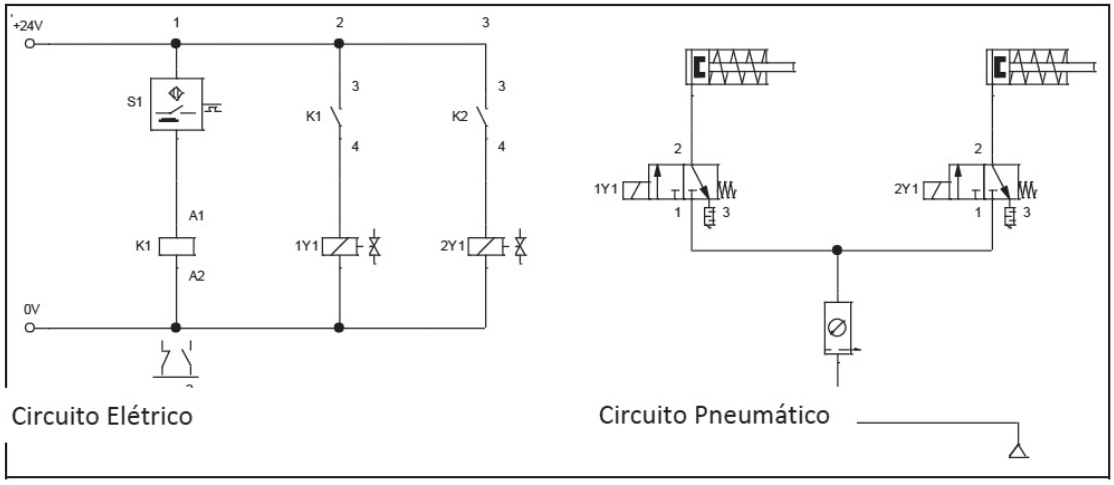
7.3.1. Descrição do circuito montado na bancada

Para facilitar as ligações na bancada utilizou-se o *software* FluidSIM, que permite representar equipamentos usados na bancada pneumática, como: válvulas, atuadores, relés, chaves e outros. A Figura 9 mostra a representação dos elementos utilizados.

O circuito elétrico começa com um sensor (S1) que aciona o avanço de dois atuadores de simples ação. A entrada do sensor (S1) foi alimentada com 24 volts. A saída do sensor foi conectada através de um cabo no relé (K1) e, depois, o relé foi conectado em 0 volts. Após isto, a chave (K1) foi alimentada com 24 volts, e a saída de (K1) foi conectada à solenoide (1Y1) e desenergizada. O sensor (S1) também é responsável por fechar o contato da chave (K2) e acionar o avanço do atuador de simples ação conectado a válvula solenoide (2Y1).

No circuito elétrico implementado foram utilizados um sensor indutivo e a placa de relés. No circuito pneumático foram utilizadas: duas válvulas solenoides de três vias e duas posições, dois atuadores de simples ação, um filtro e um suprimento de ar.

Figura 9 – Circuito eletropneumático feito no FluidSIM.



Fonte: Elaborado pelos autores (2015).

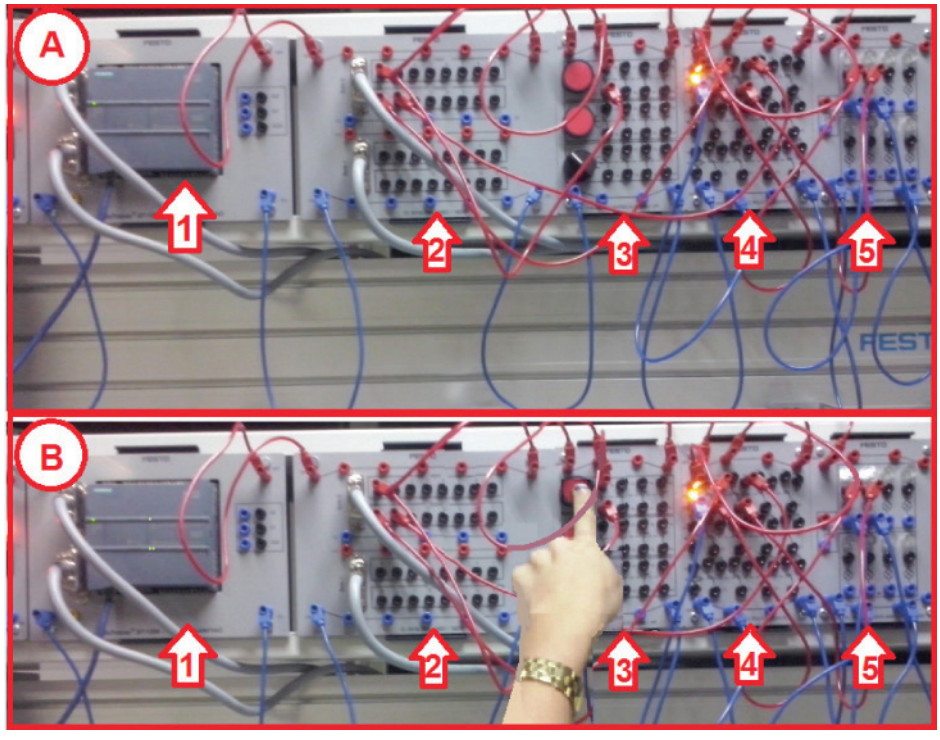
7.3.2. Montagem do circuito pneumático utilizando lâmpadas

Após ter implementado o circuito no FluidSIM foi feita a montagem do sistema na bancada. A Figura 10 mostra todas as ligações realizadas para que ocorresse o acionamento das lâmpadas.

Na Figura 10, tanto na parte superior (A) quanto na inferior (B) apresentam os mesmos elementos, porém na parte inferior (B) o botão está pressionado e as luzes acesas, conforme destacado. A seta 1 representa o módulo onde encontra-se inserido o CLP, enquanto a seta 2 mostra a placa de entradas (*inputs*) e saídas (*outputs*). Já a seta 3 representa o módulo composto por três botões, sendo um com trava. O módulo representado pela seta 4 é a placa de 3 relés e a seta 5 mostra o circuito de lâmpadas no distribuidor elétrico.

O botão que está representado pela seta 3, ao ser pressionado, emitiu um sinal de entrada para o CLP e, após 2 segundos, enviou um sinal de saída para que as lâmpadas se acendessem.

Figura 10 – Montagem do circuito na bancada pneumática. Controlador lógico programável (1), Placa de expansão de entradas e saídas (2), Placa com botões (3), Placa de 3 relés (4) e Distribuidor elétrico (5).

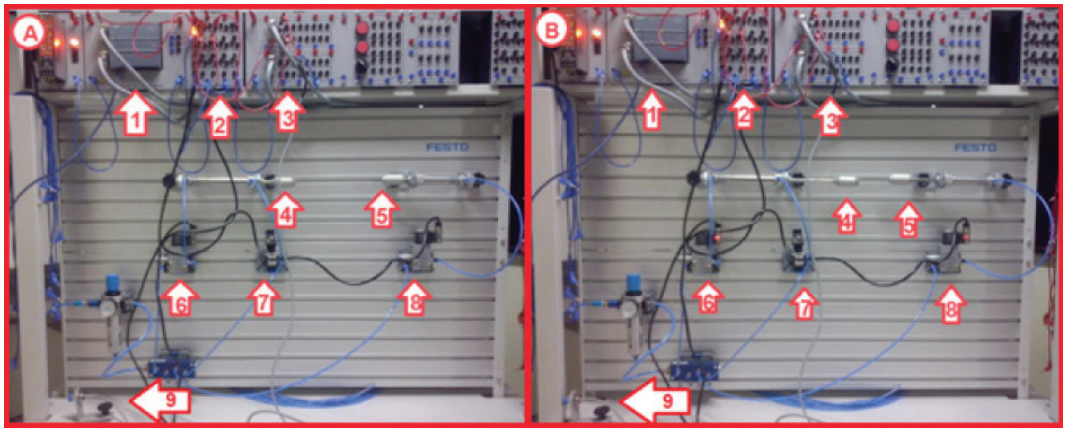


Fonte: Elaborado pelos autores (2015).

7.3.3. Montagem do circuito pneumático utilizando atuadores

Após ter elaborado o circuito no *software* FluidSIM foi realizada a montagem do sistema eletropneumático na bancada. A Figura 11 mostra todas as ligações feitas para que ocorresse o movimento dos atuadores.

Figura 11 – Montagem do circuito na bancada pneumática. Controlador lógico programável (1), Placa de 3 relés (2), Placa de expansão de entradas e saídas (3), atuadores (4) e (5), válvulas simples solenoide (6) e (8), botão giratório com trava (7) e sensor indutivo (9).



Fonte: Elaborado pelos autores (2015).

Observa-se na Figura 11 que o lado esquerdo (A) e o lado direito (B) representam o mesmo sistema, contudo há uma diferença na movimentação dos atuadores representados pela seta 4 e 5.A seta 1 representa o módulo onde encontra-se inserido o CLP, já a seta 2 mostra a placa de relés. A seta 3 representa a placa de entradas e saídas, enquanto a seta 4 mostra o cilindro de dupla ação. O acionamento que permite o avanço do cilindro é feito de forma automática através da detecção de um objeto metálico pelo sensor de proximidade indutivo mostrado pela seta 9. Sendo seu retorno executado através de um botão giratório, com trava 2/3, representado pela seta 7. A seta 5 mostra um cilindro de simples ação, o acionamento responsável pelo avanço desse atuador é feito de forma automática e ocorre quando o sensor detecta algum objeto metálico, o retorno deste atuador é feito por meio de uma mola. As setas 6 e 8 mostram as válvulas solenoides.

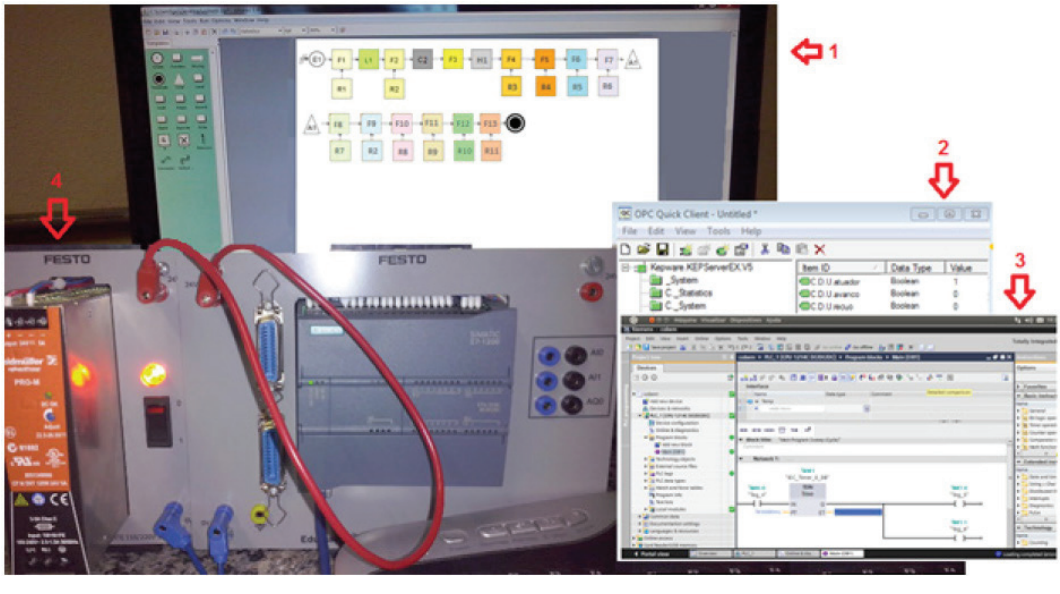
7.4. Comissionamento virtual

A segunda etapa do teste trata-se do comissionamento virtual. No teste virtual não houve necessidade da utilização da bancada pneumática. O comissionamento virtual abrangeu apenas o *software* Ururau, o CLP e o Servidor OPC. Para este fim, foram definidas variáveis no Servidor OPC, implementado o modelo de simulação no ambiente Ururau e programada a lógica de controle no CLP. O *software* Ururau gera um sinal de entrada para o CLP, e aguarda um sinal de saída. Como não se utilizou um *software* supervisório específico para visualizar o estado do sistema, as informações referentes ao funcionamento da integração foram analisadas através da variação dos valores das variáveis booleanas na interface OPC.

As variáveis configuradas iniciaram o sistema com o valor de nível lógico 0, ou seja, condição falsa, não se encontravam acionados. Pôde-se visualizar que, enquanto o simulador executava o modelo, a variável C.D.U. sensor_virtual assumia o valor de nível lógico 1, que trata-se da condição verdadeira, e após 2 segundos as variáveis C.D.U. atuador_direito e C.D.U. atuador_esquerdo recebiam o valor de nível lógico 1.

A Figura 12 apresenta o comissionamento virtual do sistema abordado, com o modelo de simulação na interface do Ururau (1), a interface do servidor OPC (2), a lógica *ladder* implementada no CLP(3) e o hardware CLP (4).

Figura 12 – Modelo de Simulação no *software* Ururau (1), Interface OPC (2), Lógica *Ladder* (3) e CLP SIEMENS S7-1200 (4).

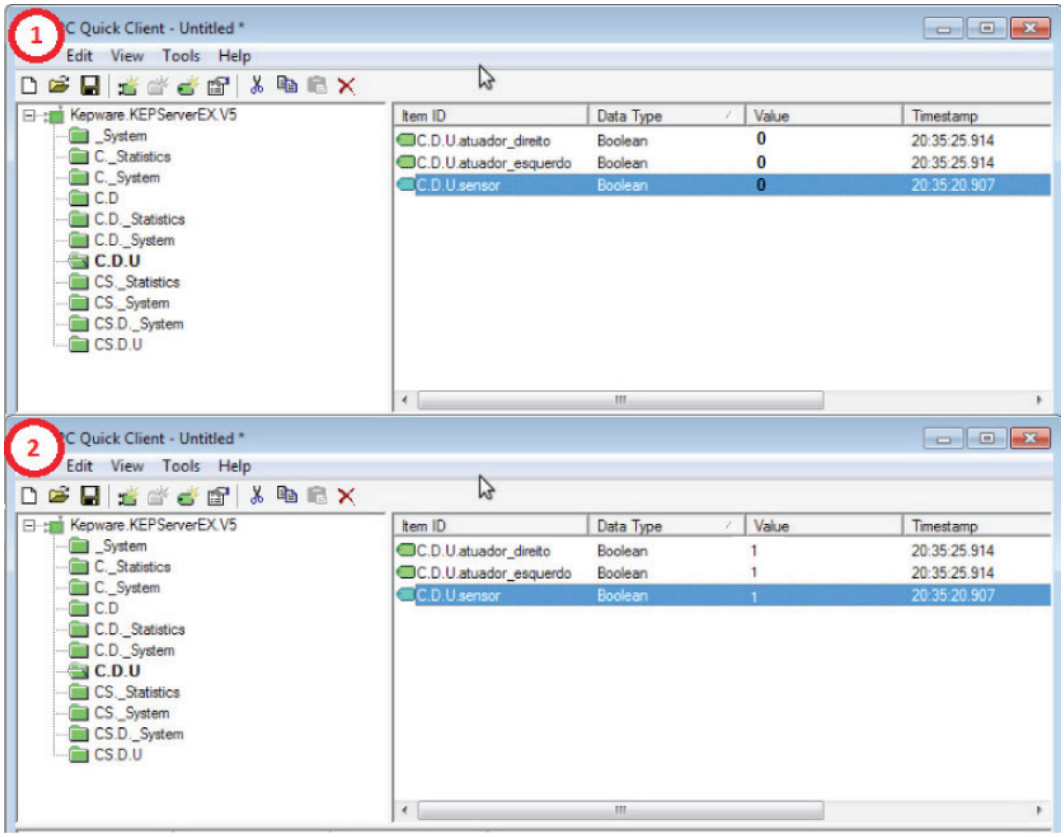


Fonte: Elaborado pelos autores (2015).

A lógica de controle feita no *software* da SIEMENS (3) utilizou um contato de entrada Normal Aberto (NA), um temporizador e duas saídas. E pode ser observada com mais detalhes na Figura 7.

Neste comissionamento apenas pôde-se visualizar a mudança das variáveis booleanas. A Figura 13 mostra a interface do Servidor OPC, através da qual se pode visualizar a mudança do estado das variáveis.

Figura 13 – Interface do Servidor OPC. Variáveis assumem valor 0 (A) e Variáveis assumem valor 1 (B).



Fonte: Elaborado pelos autores (2015).

A parte A da Figura acima mostra as três variáveis assumindo valores de nível lógico 0, ou seja, recebendo a condição falsa. A parte B mostra as variáveis assumindo valores de nível lógico 1, que é a condição verdadeira. As variáveis mudam de valor sucessivas vezes enquanto o modelo de simulação integrado ao sistema de controle estiver rodando.

7.5. Comissionamento híbrido

A terceira etapa do comissionamento é híbrida, ou seja, uma parte da planta é virtual e outra parte é física. Nesse caso utilizou-se a bancada didática, no entanto, apenas a entrada foi conectada. A Tabela 2 apresenta os endereços da entrada e das saídas usados na lógica *ladder*.

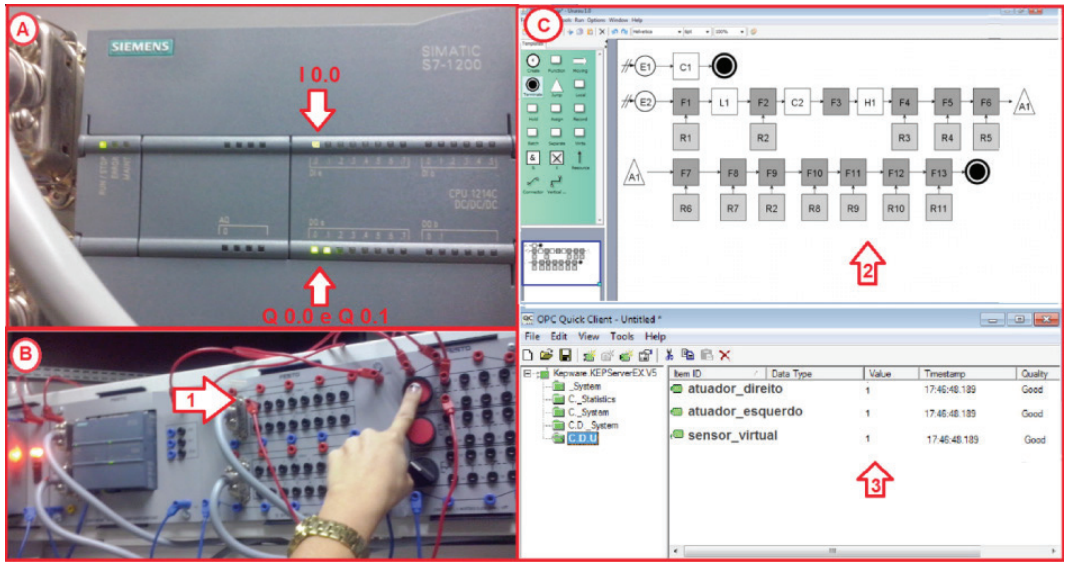
Tabela 2 – Endereço das entradas e saídas usadas na lógica *ladder*.

Símbolo	Endereço	Ação	Descrição
--] [--	I 0.0	Detectar peças	Contato NA (Normalmente aberto)
--( )--	Q 0.0	Avançar	Saída (ligar)
--( )--	Q 0.1	Avançar	Saída (ligar)

Fonte: Elaborado pelos autores (2015).

A Figura 14 mostra como foi realizado o teste híbrido. Além de visualizar o funcionamento do sistema na interface do servidor OPC (D), verificou-se o estado do sistema testado, também através dos LEDs (Light Emitting Diode) indicadores de operação no hardware CLP (A).

Figura 14 – Sistema híbrido. Controlador programável (A), Circuito elétrico (B) e Interface Ururau (C) e OPC (D).



Fonte: Elaborado pelos autores (2015).

Quando o botão presente na placa de botões de comando elétrico na parte B da Figura acima foi pressionado, foi enviado um sinal de entrada digital para o CLP. O LED mostrado na parte A da Figura acima, acendeu, comunicando que foi feita a leitura do dado de entrada (I 0.0).

Após 2 segundos, devido ao temporizador inserido na programação do circuito de controle, os dois LEDs de saída mostrados na parte A da Figura acima acenderam (Q 0.0, Q 0.1) informando que o CLP gerou o sinal de saída. A seta 1 mostrada na parte B da figura acima representa a alimentação da entrada digital, com endereço I0.0.

Simultaneamente a visualização dos LEDs, pôde-se observar as variáveis na parte C da Figura acima, assumindo os valores de nível lógico 1 e 0 durante a execução do modelo de simulação (C), assim como foi mostrado no comissionamento virtual.

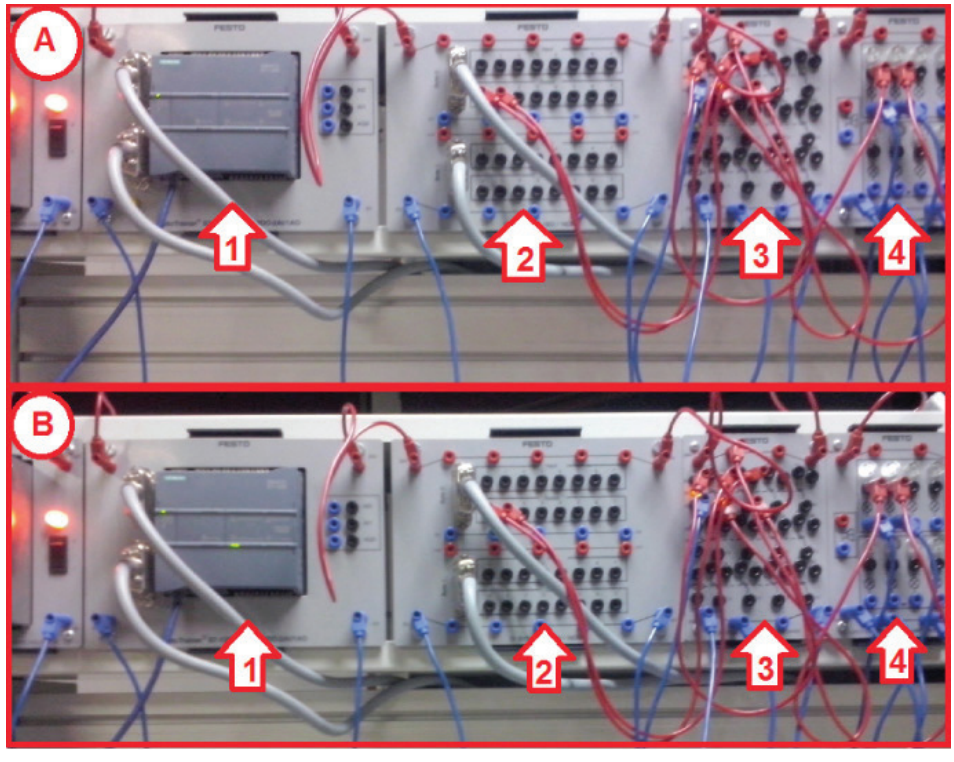
7.6. Comissionamento *reality in the loop*

O comissionamento *Reality in the Loop* foi realizado utilizando a planta real (bancada didática) controlada virtualmente. O sensor foi simulado para gerar um sinal de entrada para o sistema. Sendo assim, a entrada do sistema foi feita através do acesso a memória do *software* Ururau que enviou um sinal de entrada para o CLP. O controlador leu esse dado e escreveu a saída acionando os elementos da bancada, que, no primeiro momento, foram duas lâmpadas e, posteriormente, dois atuadores, como será mostrado nas próximas seções.

7.6.1. Comissionamento *reality in the loop* utilizando lâmpadas

O Comissionamento *Reality in the Loop* utilizando Lâmpadas pode ser observado na Figura 15. Nesta Figura tanto a parte superior (A) quanto à parte inferior (B) representam o mesmo sistema. Contudo, a parte A da Figura mostra o distribuidor elétrico, representado pela seta 4, com as 8 lâmpadas desligadas enquanto a parte B mostra duas lâmpadas ligadas.

Figura 15 – Comissionamento *Reality in the Loop* utilizando lâmpadas. Controlador programável (1), Placa de expansão de entradas e saídas (2), Placa de 3 relés (3) e Distribuidor elétrico (4).



Fonte: Elaborado pelos autores (2015).

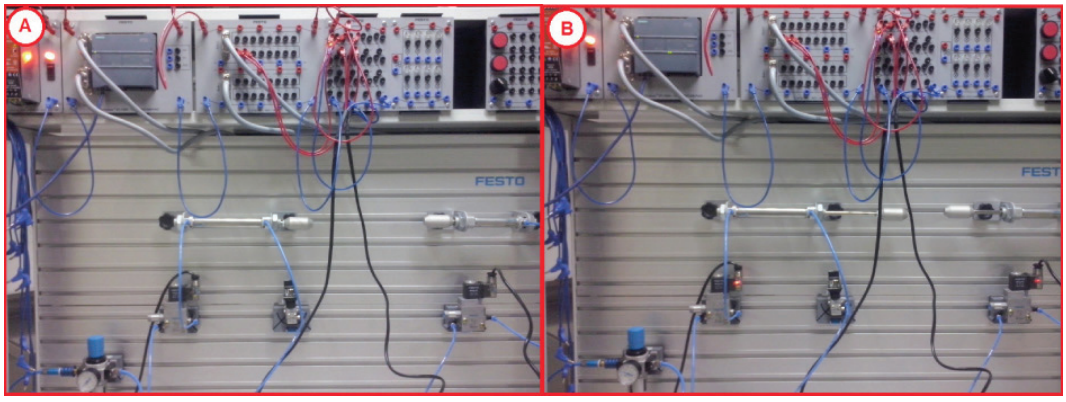
No início da execução do modelo de simulação, a variável C.D.U. sensor virtual permaneceu com nível lógico 0, de modo que as saídas não foram acionadas (A). Após 2 segundos o controlador enviou o sinal de saída e duas lâmpadas acenderam no distribuidor elétrico (B), que indica que a variável C.D.U. sensor virtual assumiu nível lógico 1.

Nota-se ainda, na Figura 15, que a entrada na placa de expansão representada pela seta 2 não foi alimentada, conforme circulado, apenas as saídas.

7.6.2. Comissionamento *reality in the loop* utilizando atuadores

O Comissionamento *Reality in the Loop* utilizando atuadores pode ser observado na Figura 16. Nesta Figura tanto a parte esquerda (A) quanto à parte direita (B) representam o mesmo sistema. Porém, apresentam diferença em ralação ao avanço dos atuadores, circulados em branco.

Figura 16 – Comissionamento *Reality in the loop* utilizando atuadores. Atuadores com o recuo acionado (A) e Atuadores com avanço acionado (B).



Fonte: Elaborado pelos autores (2015).

No princípio da execução do modelo de simulação, a variável C.D.U. sensor virtual assumiu o valor de nível lógico 0 de modo que, as saídas não foram acionadas, parte A da Figura.

Após alguns segundos, o avanço dos atuadores foi acionado e isto pode ser visto na parte B da Figura 16 que significa que, a variável C.D.U. sensor virtual assumiu nível lógico 1. Assim, o controlador enviou o sinal de saída após 2 segundos da leitura do sinal de entrada, devido ao temporizador, fazendo com que o movimento dos atuadores fosse acionado.

8. CONSIDERAÇÕES FINAIS

O presente trabalho descreveu o mecanismo de integração entre um modelo de simulação e um sistema de controle utilizando o *software* de simulação a eventos discretos Ururau. Para isso foi construído um modelo de simulação de uma unidade industrial hipotética com a finalidade de demonstrar a realização de um possível comissionamento em uma parte do sistema modelado. Foram demonstrados também diferentes alternativas para procedimentos que podem ser aplicados durante a realização de diversas etapas de um comissionamento.

O trabalho também buscou demonstrar conceitos relacionados a comissionamentos de sistemas de controle. Além disso, buscou demonstrar ainda o procedimento de como o mecanismo de integração dos módulos do *software* Ururau se integram com um controlador digital e algumas características gerais do respectivo *software*. Foi descrito o modelo de simulação e indicado os parâmetros que possibilitam a construção do mesmo. Os resultados buscaram ilustrar o funcionamento do sistema de controle e a configuração da interface de comunicação.

Foram realizados diversos testes de forma a demonstrar a capacidade do novo *software* realizar diferentes procedimentos para uns comissionamentos. Assim, foi demonstrado que o *software* livre Ururau permite o comissionamento de um sistema de controle integrado a um modelo de simulação. Sugere-se, para trabalhos futuros, comparar o desempenho do *software* Ururau com *softwares* específicos de comissionamento e utilizar o Ururau para realização de comissionamento em sistemas reais. Por fim, destaca-se que o projeto do *software* Ururau busca ampliar a fronteira em relação ao uso da simulação discreta no Brasil.

9. AGRADECIMENTOS

Os autores gostariam de agradecer a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPQ) e a Fundação de Amparo à Pesquisa do Estado do Rio de Janeiro (FAPERJ) pelo suporte financeiro para esta pesquisa.

REFERÊNCIAS

AUINGER, F.; VORDERWINKLER, M.; BUCHTELA, G. Interface driven domain-independent modeling architecture for "Soft-Comissioning" and "Reality in the Loop". *In*: WINTER SIMULATION CONFERENCE, 805, 1999. **Anais...** Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, 1999.

BANKS, J. Simulation in the Future. *In*: WINTER SIMULATION CONFERENCE, 9-16, 2000. **Anais...** Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, 2000.

BANKS J.; CARSON J. S.; NELSON B. L.; NICOL D. **Discrete-event System Simulation**, 5th ed. Prentice-Hall: Englewood Cliffs, NJ, 2010.

CARDOSO, L. D.; RANGEL, J. J. A.; BASTOS, P. J. T. Ambiente Híbrido Utilizando Simulação a Eventos Discretos e Células de Manufatura para Desenvolvimento e Teste de Sistemas de Controle. *In*: CONGRESSO BRASILEIRO DE AUTOMÁTICA, 19, p. 1-7, 2012. **Anais...** Campina Grande, 2012. Disponível em: <http://www.eletrica.ufpr.br/anais/cba/2012/Artigos/99997.pdf>. Acesso em: 05 jan. 2016.

CARDOSO, L. D.; RANGEL, J. J. A.; BASTOS, P. J. T. Discrete event simulation for integrated design in the production and commissioning of manufacturing systems. *In*: WINTER SIMULATION CONFERENCE, 2544-2552, 2013. **Anais...** Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, 2013.

CARLSSON, H.; SVENSSON, B.; DANIELSSON, F.; LENNARTSON, B. Methods for reliable simulation-based PLC code verification. Industrial Informatics. **IEEE Transactions On**, v. 8, n. 2, p. 267-278, 2012.

DAGKAKIS, G.; HEAVEY, C. A review of open source discrete event simulation software for operations research. **Journal of Simulation**, v. 10, n. 3, p. 193-206, 2016.

DOMINKA, S.; BENDER, K. **Hybrid plant startup:** from the hardware in the loop simulation to the actual facility. University College of Southeast Norway, v. 102, n. 1, p. 25-31, 2007.

DOUGALL, D. J. Applications and benefits of real-Time I/O simulation for PLC and PC control systems. **ISA Transactions**, v. 36, n. 4, p. 305-311, 1998.

JAIN, S.; LINDSKOG, E.; JOHANSSON, B. Supply chain carbon footprint tradeoffs using simulation. *In*: WINTER SIMULATION CONFERENCE, 3168-3179, 2012. **Anais...** Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, 2012.

KING D.; HARRISON H. Discrete-event simulation in Java: A practitioner's experience. *In*: CONFERENCE ON GRAND CHALLENGES IN MODELING AND SIMULATION, SOCIETY FOR MODELING AND SIMULATION: INTERNATIONAL VISTA. p. 436–441, 2010. **Anais...** CA, 2010.

KO, M.; PARK, S. C. Template-based modeling methodology of a virtual plant for virtual commissioning. **Concurrent Engineering**, v. 22, n. 3, p. 197-205, 2014.

MONTEVECHI, J. A. B.; LEAL, F.; PINHO, A. F.; COSTA, R. F. S.; OLIVEIRA, M. L. M.; SILVA, A. L. F. Conceptual modeling in simulation projects by mean adapted IDEF: An application in a Brazilian tech company. *In*: WINTER SIMULATION CONFERENCE, 1624-1635, 2010. **Anais...** Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, 2010.

PEIXOTO, T. A.; RANGEL, J. J. A.; MATIAS, I. O. Usando o JSL para Simulação de Monte Carlo. **Revista Gestão da Produção, Operações e Sistemas**, v. 4, p. 135-152, 2012.

PEIXOTO, T. A.; RANGEL, J. J. A.; MATIAS, I. O. Free and Open-Source Simulation Software "Ururau". *In*: SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL, 47, 2015. **Anais...** Porto de Galinhas - PE, v. 1, p. 1-12, 2015.

RANGEL, J. J. D. A.; SOUZA, A. A.; BASTOS, P. J. T.; BAPTISTA, R. C. T. Simulação a eventos discretos para treinamento em sistemas de controle. **Pesquisa Operacional para o Desenvolvimento**, v. 4, n. 1, p. 97-111, 2012.

ROSSETI, M. D. Java Simulation Library (JSL): An Open-Source Object-Oriented Library for Discrete-Event Simulation in Java. **International Journal of Simulation and Process Modelling**, v. 4, p. 69-87, 2008.

SAKURADA, N.; MIYAKE, D I. Aplicação de simuladores de eventos discretos no processo de modelagem de sistemas de operações de serviços. **Revista Gestão e Produção**, v. 16, p. 25-43, 2009.

SARGENT, R. G. Verifications and validation of simulations models. **Journal of Simulation**, v. 7, p. 12–24, 2013.

SMITH, J. S.; CHO, Y. Offline commissioning of a plc-based control system using arena. *In*: WINTER SIMULATION CONFERENCE, 1802-1810, 2008. **Anais...** Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, 2008.

SWAIN, J. J. Discrete Event Simulation Software Tools: A better reality. **OR/MS Today**, v. 40, n. 5, p. 48-59, 2013.

VERSTEEGT, C.; A. VERBRAECK. The Extended Use of Simulation in Evaluating Real-Time Control Systems of AGVs and Automated Material Handling Systems. *In*: WINTER SIMULATION CONFERENCE, 1659-1666, 2002. **Anais...** Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, 2002.