Gregório Juodinis Santos e Heitor de Carvalho

Projeto e desenvolvimento de interface tridimensional para sistema supervisório e de treinamento

São Paulo

2016

Gregório Juodinis Santos e Heitor de Carvalho

Projeto e desenvolvimento de interface tridimensional para sistema supervisório e de treinamento

Trabalho de formatura submetido à Escola Politécnica da Universidade de São Paulo como requisito para a obtenção de diploma de graduação em Engenharia Mecatrônica.

Universidade de São Paulo – USP

Escola Politécnica

Departamento de Engenharia Mecatrônica

Orientador: Prof. Dr. Fabrício Junqueira

São Paulo

2016

Gregório Juodinis Santos e Heitor de Carvalho
Projeto e desenvolvimento de interface tridimensional para sistema supervisório e de treinamento/ Gregório Juodinis Santos e Heitor de Carvalho. – São Paulo, 2016
Orientador: Prof. Dr. Fabrício Junqueira

Trabalho de formatura – Universidade de São Paulo – USP
Escola Politécnica
Departamento de Engenharia Mecatrônica, 2016.

CDU 02:141:005.7

Agradecimentos

Os autores gostariam de agradecer ao orientador Prof. Dr. Fabrício Junqueira pelo imenso apoio ao longo de um processo que durou mais de um ano e que se mostrou um imenso e engrandecedor desafio. Além disso, gostaríamos também de agradecer ao Dr. Marcosíris Amorim Pessoa, por estar sempre diposto a nos ajudar e torcendo pela nova evolução.

Precisamos, também, fazer alguns agradecimentos pessoais:

Em mais que merecido primeiro lugar, à família. Aqueles que me trouxeram até aqui e se orgulham em ver cada passo que eu dou. Em especial, mãe, pai, Lucas, Rafa, Sessa, vó Geny, vô Roberto, vó Maria e vô Zé: muito obrigado! Aos amigos, por serem os companheiros de luta, pelas boas conversas e pelos bons conselhos: muito obrigado! À Isa, por ser aquela pra quem eu tive orgulho de mostrar os progressos e liberdade pra dividir as falhas: muito obrigado! Ao Heitor, por ser o parceiro que se manteve firme até o fim junto comigo nessa que foi uma batalha muito maior do que qualquer um de nós esperava: muito obrigado! Aos professores que me acompanharam até aqui, por dedicarem uma parte de si a me tornar um estudante e uma pessoa melhor: muito obrigado! A todos que de alguma forma contribuíram para que eu fosse quem sou hoje: muito obrigado! Fica aqui a minha palavra de que eu vou fazer valer a pena todo o esforço e confiança que vocês depositaram em mim!

Com carinho, Gregório

Das várias pessoas que passaram ou estão presentes em minha vida, gostaria de agradecê-las pelos mais variados motivos, sendo elas: à minha família, com foco especial nos meus pais, Rubens e Telma, e meus irmãos, Thomaz e Enzo, que sempre me apoiarão e que nunca hesitarão em lutar por mim, sendo que tenho certeza que os deixarei orgulhosos com as diretrizes que tenho definido em minha vida; Aos meus amigos, por serem grandes companheiros, que compartilharam momentos de alegria e auxiliaram em momentos de baixa; Ao Gregório, que foi de fundamental importância na concepção desse projeto e que, independente das adversidades durante o ano, sempre esteve ao meu lado, sendo além da minha dupla de trabalho de conclusão, um grande amigo; E um obrigado também à todos que de alguma forma agregaram também em algo na minha formação profissional, acadêmica e pessoal.

Com grande apreço, Heitor

Resumo

A forma com que o homem se relaciona com a máquina e o meio virtual evolui de acordo com o conhecimento tecnológico e o desejo de criar novas formas de interação, sendo um aspecto importante tanto no aprendizado quanto na operação de ferramentas virtuais e digitais. O presente trabalho tem como objetivo a obtenção de um sistema integrado de supervisão e treinamento para planta controlada por CLP, que introduz novas funcionalidades por meio da aplicação de elementos de Realidade Virtual. O sistema desenvolvido usa esses elementos para permitir que o usuário interaja com o sistema de forma intuitiva e sem riscos, além de possibilitar a conexão com o sistema real para supervisão do sistema e habilitação do seu controle.

Palavras-chaves: Sistemas Supervisórios, Realidade Virtual, Protocolos OPC, Automação Industrial

Abstract

The way that man interacts with machinary and the virtual environment develops according to the technological knowledge and the desire to create new forms of it, being highlighted as an important aspect both in learning and operation of digital and virtual tools. The present work targets to obtain an integrated supervision and training system for CLP controlled plant, which introduces new functionalities through the application of elements brought from the field of Virtual Reality. The developed system uses these elements to allow the user to interact with it in an intuitive and unharmful manner, as well as enabling the connection with the real system for its supervision and control.

Key-words: Supervisory Systems, Virtual Reality, OPC Protocol, Industrial Automation

Lista de ilustrações

Lista de abreviaturas e siglas

CLP	Controlador Lógico Programável
RV	Realidade Virtual
SCADA	*Supervisory Control and Data Acquisition*
IHM	Interface Homem-Máquina
RTU	*Remote Terminal Unit*
OPC	*OLE for Process Control*
COM	*Component Object Model*
DCOM	*Distributed Component Object Model*
RPC	*Remote Procedure Call*
IEC	*International Electrotechnical Commission*
SFC	*Sequential Function Chart*
TE	Texto Estruturado
POU	*Program Organization Unit*
CPU	*Central Processing Unit*
SR	Sistema Real
MV	Modelo Virtual
CAD	*Computer Aided Design*
FBX	*Filmbox*
SDK	*Software Development Kit*
UML	*Unified Modelling Language*
MT	Modo Teste
MS	Modo Supervisório

Sumário

1 Introdução

A indústria habilita o homem a amplificar sua capacidade de transformação dos bens naturais e os avanços tecnológicos têm atuado na expansão dessa capacidade desde a Revolução Industrial (RÜSSMANN et al., 2015). Os fatores que levam à inovação na indústria podem ser divididos em duas forças principais: o *pull* da demanda e o *push* dos avanços tecnológicos. O primeiro inclui a redução no período de desenvolvimento de produtos, fator de sucesso para companhias de bens de consumo; a individualização da demanda, com cada vez mais liberdade de customização por parte do consumidor; e o esforço por racionalização do uso de recursos naturais. Já o segundo, em especial ao longo dos últimos anos, é o resultado das tendências tecnológicas de automação, digitalização e integração dos sistemas físicos com os computacionais (LASI et al., 2014). Ainda segundo Lasi *et al.* (2014), a digitalização das ferramentas de suporte à manufatura tem como uma de suas consequências o avanço de tecnologias de simulação, segurança digital e realidade virtual.

A realidade virtual (RV), além da aplicação direta na indústria dos jogos de entretenimento e, ainda mais, dos jogos com caráter funcional (ZYDA, 2005), traz novas perspectivas para a manufatura. A Ford, uma montadora historicamente conhecida por aplicar inovação em sua linha de montagem, já reconheceu a importância dessa tendência tecnológica e desenvolveu, em colaboração com a Siemens, uma ferramenta para visualização tridimensional de suas plantas por meio da integração com informações do *Google Earth* (FORBES, 2014). Com a ferramenta, chamada *IntoSite* (Figura 1), os engenheiros da Ford podem encontrar eventuais problemas no processo de fabricação mais facilmente e com maior eficiência, de acordo com Janice Goral - Gerente de Engenharia de Manufatura na montadora.

Para Roher (ROHRER, 2000), a representação gráfica e animação dos objetos físicos é essencial na simulação de processos de manufatura, já que o ser humano naturalmente tem mais facilidade para capturar e processar informações no formato visual. O autor afirma que diversos aspectos de um modelo de manufatura tem uma tradução direta para o meio gráfico, enquanto que isso não aconteceria com a linguagem textual, por exemplo. Desta forma, a animação na simulação virtual de um processo de manufatura pode trazer benefícios, em particular na: validação e verificação do modelo; compreensão e comunicação dos resultados; e obtenção de credibilidade.

Tendo em vista a facilidade do ser humano em interagir com a informação no formato visual, torna-se relevante considerar as vantagens que a RV poderia trazer ao ser introduzida na interface entre o homem e os processos produtivos nos sistemas supervisórios,

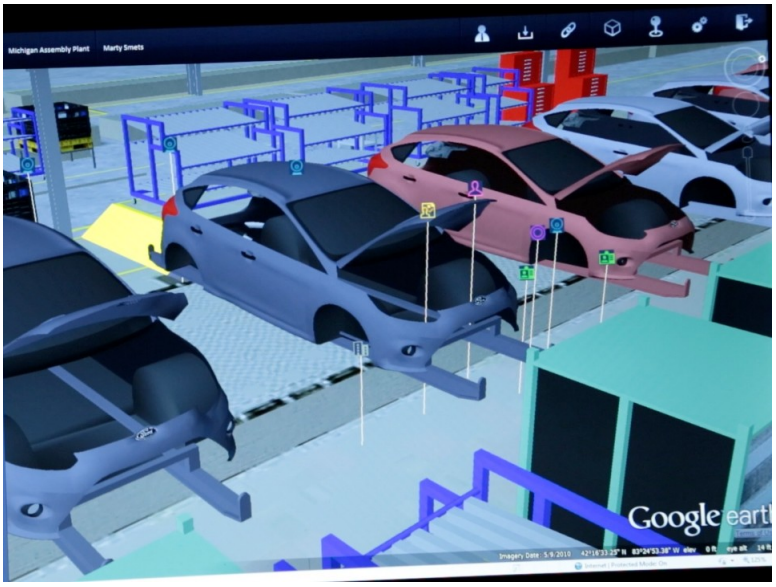

Figura 1 – Exemplo de visualização do *IntoSite* da Ford, com modelo tridimensional virtual de uma planta de produção. Retirado de (FORBES, 2014)

ou sistemas SCADA (do inglês, *Supervisory Control and Data Acquisition* - Sistema de Supervisão e Aquisição de Dados).

Usualmente, o que se encontra em soluções de mercado dos principais nomes da automação é uma representação gráfica bidimensional com caráter ilustrativo, sem um paralelo direto com o mecanismo real que está sendo acompanhado. Exemplo disso é o sistema fornecido pela ABB, ilustrados na Figura 2.

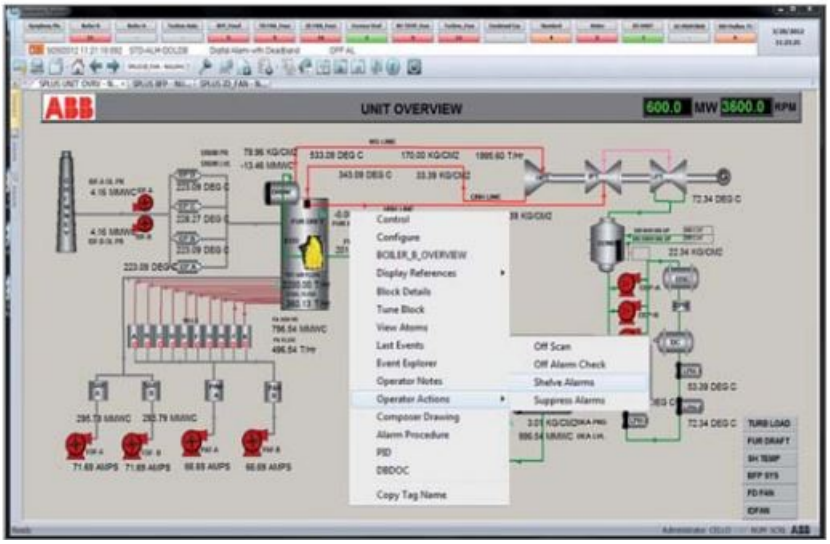

Figura 2 – Exemplo de tela do sistema *Symphony Plus* da ABB (ABB, 2012)

A complementação de um desses sistemas com o modelo virtual tridimensional dos mecanismos supervisionados e a animação destes de acordo com o estado dinâmico da planta poderia, além de trazer um nível adicional de informação, facilitar a tradução do

que é mostrado na interface com o usuário para a tomada de decisão dentro do ambiente fabril.

Outro aspecto a ser considerado, e que está diretamente ligado à experiência dos autores em sua trajetória ao longo da graduação em engenharia, é a capacitação para operação de ferramentas tecnológicas de apoio à produção. Um exemplo é o de sistemas controlados por CLP (Controlador Lógico Programável), em que é necessário traduzir um processo que se deseja implementar em uma sequência de regras lógicas com as entradas e saídas (sensores e atuadores) disponíveis. Por fim, deve-se redigir um programa em linguagem que seja aceita neste tipo de sistema e verificar se o processo é executado conforme esperado.

Um modelo de RV da planta pode ser um passo intermediário na tradução de um processo especificado em linguagem natural para um programa que possa ser carregado no CLP. A interação com o modelo virtual valeria para que o operador compreendesse como cada um dos componentes do sistema real reagem a uma ação desejada, como, por exemplo, o ativamento de uma esteira de transporte em uma linha de montagem. Com isso, é direta a visualização de qual é o estado do sistema a cada passo do processo e quais são os critérios de decisão no sequenciamento lógico do programa que deve ser implementado.

As duas vantagens principais desta abordagem são: I) a possibilidade de se desconectar da planta real para realizar testes, o que elimina o risco de danificar os mecanismos ou até mesmo de ferir o operador; e II) a utilização de uma interface de atuação próxima da linguagem natural (com atuação de cada um dos componentes em separado), em oposição ao ciclo tradicional de programação, carregamento no CLP e depuração.

O presente trabalho visa criar uma interface tridimensional para sistema supervisório, com o intuito de demonstrar o conceito por meio de um estudo de caso. Além disso, esse sistema deve possuir um modo de uso de treinamento que possibilite a operação de um modelo virtual do sistema real. Mais especificamente, o caso a ser estudado é o de uma bancada educacional que simula um ambiente de manufatura em pequena escala.

2 Objetivo

O objetivo deste projeto é a obtenção de um sistema integrado de supervisão e treinamento com interface em realidade virtual. Este sistema foi composto por um modelo virtual, em programa desenvolvido pelos autores, e um sistema real, para o qual será utilizada uma solução pronta de mercado. O modelo deve reproduzir os mecanismos em sua forma e escala real, e ser animado de acordo com os estados de atuadores e sensores do CLP do sistema real.

O modelo virtual deve conter uma Interface Homem-Máquina (IHM), uma réplica tri-dimensional do mecanismo, e uma interface de comunicação com o controlador. Este modelo poderá ser operado tanto de forma isolada do sistema real, a partir de comandos do usuário, quanto integrado a este, seguindo o estado dos componentes conforme forem processados no CLP ou enviando comandos de atuação. Está fora do escopo o tratamento de eventos externos ao processo (e.g. a retirada de uma peça da linha por ator externo); e a programação, carregamento ou interpretação de código para CLP.

O sistema real é composto por uma bancada educacional que simula um processo produtivo em pequena escala e um CLP. Está fora do escopo do projeto o desenvolvimento destes componentes, que serão integrados ao sistema em sua configuração comercial, e o tratamento de qualquer informação que não possa ser capturada com os sensores existentes. A arquitetura do sistema será abordada em detalhes em seções posteriores.

3 Revisão Bibliográfica

3.1 Realidade Virtual

A Realidade Virtual (RV) pode ser definida como a aplicação de um sistema computacional na criação de uma interface interativa e tridimensional, em que os objetos reais são representados com sua forma espacial (HAMID; AZIZ; AZIZI, 2014). Em (MUJBER; SZECSI; HASHMI, 2004), os autores citam três áreas para aplicação da RV na manufatura: usinagem, montagem e inspeção. Neste contexto, o uso da RV está definido pela aplicação no estudo das relações entre peça e ferramenta no trabalho de remoção de material; a relação entre as diferentes partes de um sistema produtivo, incluindo peças de trabalho e atuadores; e a simulação da inspeção de propriedades físicas e mecânicas dessas partes.

Existem três tipos principais de sistemas de realidade virtual, que se diferenciam pelo grau de imersão do usuário: os não imersivos, os semi-imersivos e os totalmente imersivos (MUJBER; SZECSI; HASHMI, 2004). O primeiro tipo abrange os sistemas cuja interface é a tela de um computador e a interação com o usuário é feita por meio de *mouse* e teclado. No outro extremo, existem sistemas que utilizam aparatos como *smartglasses* e sensores de voz e movimento para criar a sensação de completa imersão do usuário no sistema, como o que é mostrado na Figura 3. O foco deste trabalho é o primeiro tipo, visto que o escopo do *software* aqui desenvolvido não inclui a interação do usuário por meio de um ambiente de imersão e responsividade a comandos de voz e movimento.

Figura 3 – Exemplo de interação de usuário com sistema de RV totalmente imersivo. Retirado de (MUJBER; SZECSI; HASHMI, 2004)

Em (LATTA; OBERG, 1994), A RV é definida como uma interface entre homem e máquina que simula realisticamente um ambiente real e permite que o usuário interaja com este ambiente simulado. Para os autores, o grau de realismo é dificilmente colocado

em termos quantitativos, mas deve ser suficiente para que o usuário do sistema em questão consiga perceber a equivalência entre o ambiente simulado e o real.

Ademais, os autores definem um sistema de RV operacional como um em que o usuário pode executar operações que não seriam facilmente executadas de outra forma, citando o exemplo de um simulador de voo em que uma pessoa destreinada pode ter uma experiência próxima do controle de um avião sem riscos consideráveis. Neste mesmo contexto, o presente trabalho permite que um usuário não familiarizado com o ambiente de produção simulado possa interagir com um modelo virtual e experimentar diferentes operações sem a necessidade da conexão com o sistema real.

Ainda no âmbito de utilização de RV para aprendizagem, um dos casos apresentados em (LOEFFLER; ANDERSON, 1994) é o *Explore Net*, um ambiente de RV que apresenta um mundo simulado no qual os estudantes possuem réplicas virtuais com as quais entram em situações que estimulam o desenvolvimento de habilidades cognitivas e de interação com outros participantes. Tal qual o sistema aqui proposto, o *Explore Net* (Figura 4) apresenta uma forma limitada de imersão física. No entanto, os autores E. Hughes e J. Moshell argumentam que o propósito educacional não é necessariamente atingido por um alto grau de imersão ou por imagens estereoscópicas, mas sim pela capacidade da realidade virtual de permitir a interação e exploração por parte usuário. Esse é um grau de liberdade que se intencionou fornecer aqui, já que o usuário pode atuar no sistema comandando livremente cada um dos mecanismos, em oposição à necessidade de se construir um programa que é executado pelo CLP (Controlador Lógico Programável) no sistema real.

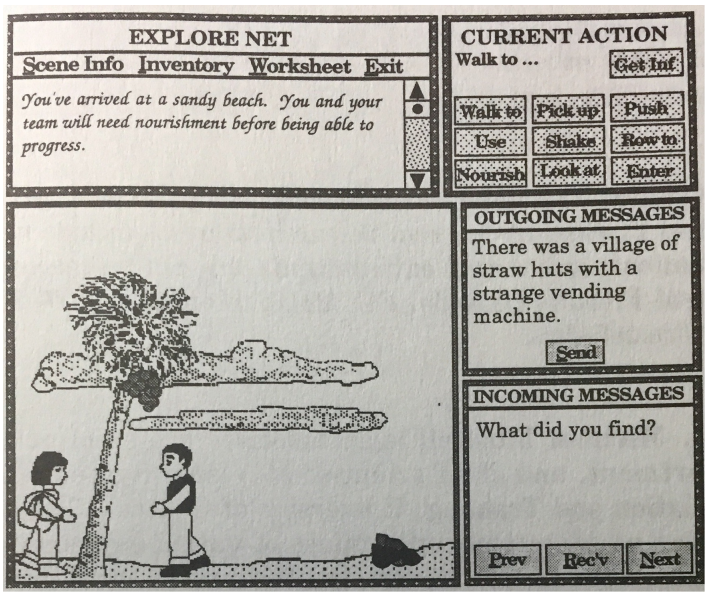

Figura 4 – Tela do programa educacional *ExploreNet*. Retirado de (LOEFFLER; ANDERSON, 1994)

Outro conceito ligado à RV é a presença, também chamada de telepresença. Em (STEUER, 1992), o autor argumenta que definir a RV como um conjunto de aparatos

tecnológicos como os óculos de realidade aumentada com monitor integrado ou luvas com sensor tátil não é suficiente, já que implicaria que a RV está limitada ao ciclo de vida e capacidades dessas tecnologias.

Sendo assim, o autor exalta a importância em sistemas de RV da presença: a forma como o usuário experiencia seu ambiente e arredores, a sensação de se estar em um ambiente mesmo que por mediação de componentes virtuais. Ainda de acordo com o autor, quando o usuário tem sua interação com o ambiente intermediada por algum meio, dois ambiente se tornam presentes: o real e o virtual. A telepresença se dá quando o usuário favorece o segundo em detrimento do primeiro, ou seja, quando o ambiente com o qual ele interage é o criado pelo meio de intermediação.

No presente projeto, espera-se que o operador possa ter a experiência de telepresença ao supervisionar sua planta à distância. Ao avaliar a condição das peças e mecanismos na interface, o usuário deveria ter a sensação de estar observando a própria planta em operação ainda que esteja distante do sistema real. Desta forma, ele deve direcionar suas ações da mesma forma que faria se estivesse supervisionando a planta pessoalmente.

3.1.1 Exemplos de aplicação da Realidade Virtual

A RV é usada de várias maneiras, incluindo projeto, simulação, treinamento e venda, com o objetivo de aumentar a eficiência e reduzir os custos dos processos em que é empregada (NETTO; MACHADO; OLIVEIRA, 2002).

Na indústria, a RV já vem sendo usada para a prototipagem, em detrimento da construção de protótipos físicos, por empresas como Caterpillar e General Motors; na avaliação de novos *layouts* de fábrica pela BMW; para treinamento de funcionários em novos processos pela Rolls Royce Aeroengines and Associates; e para o projeto de aeronaves pela Embraer (PORTO et al., 2002).

Segundo (NETTO; MACHADO; OLIVEIRA, 2002), outras aplicações incluem:

- Projetos arquitetônicos, em especial na substituição de maquetes tradicionais para modelagem e venda de novos empreendimentos.
- Tratamento de pacientes com fobia, permitindo que estes entrem em contato com situações que desencadeiam a reação de fobia em um ambiente com maior possibilidade de controle por parte de médicos e terapeutas.
- Treinamento de cirurgiões para procedimentos de alta exigência de habilidade manual, como a cirurgia laparoscópica.
- Validação de montagem de componentes com partes virtuais e avaliação de colisão, em particular na indústria automotiva.

3.1.2 Desenvolvimento de sistema em RV

Os autores em (NETTO; MACHADO; OLIVEIRA, 2002) propõem um procedimento para projeto de sistema em RV, que é descrito em suas etapas na Figura 5.

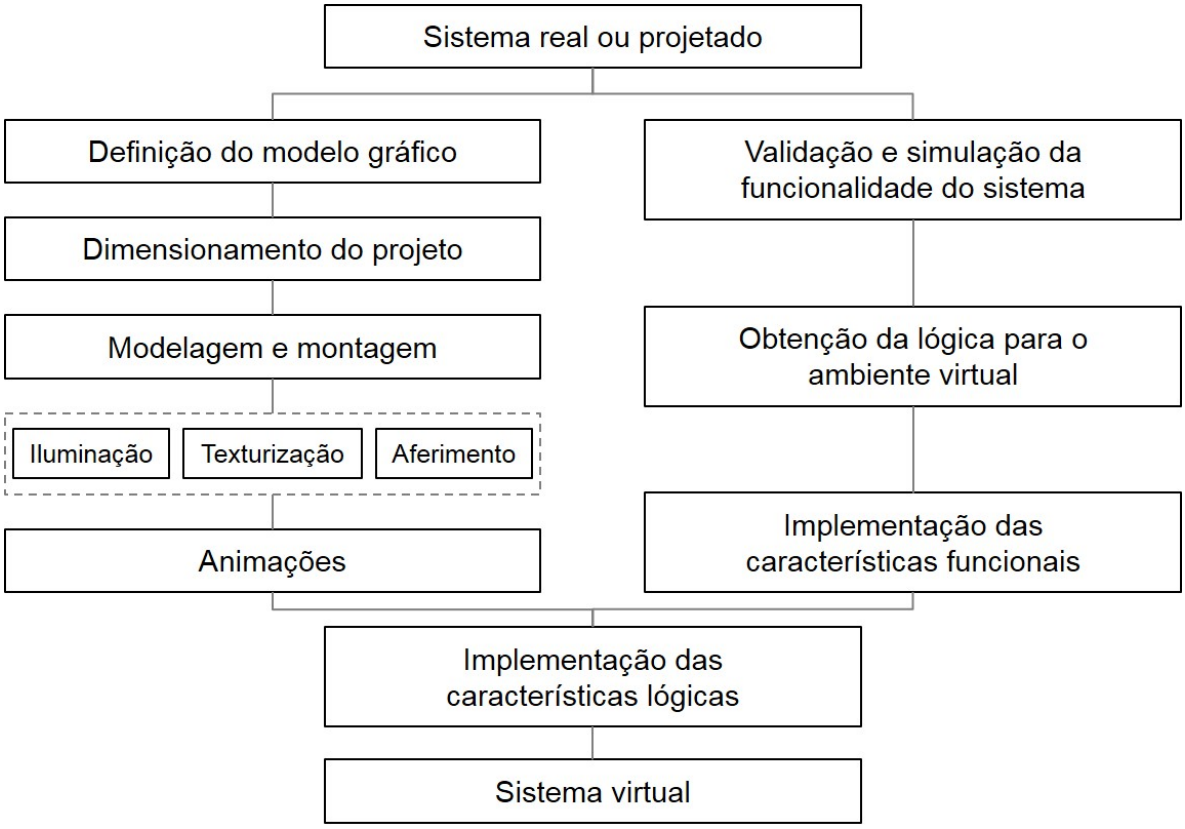

Figura 5 – Procedimento para desenvolvimento de sistema em RV. Adaptado de (NETTO; MACHADO; OLIVEIRA, 2002)

O procedimento tem duas cadeias principais de etapas que são executadas em paralelo e que se unem na implementação das características lógicas do modelo: I) à esquerda, etapas de desenvolvimento gráfico e geométrico do projeto; e II) à direita, etapas de validação e simulação das funcionalidades do sistema.

No primeiro grupo, deve-se modelar as características gráficas do sistema em estudo de forma a permitir a sua utilização em dispositivos de processamento gráfico, atentando ao equilíbrio entre a complexidade do modelo e o aumento na utilização de recursos de processamento durante a operação do sistema.

No projeto aqui proposto, essas etapas foram seguidas com o uso de *softwares* de modelagem computadorizada para se obter o modelo tridimensional da bancada educacional (objeto de estudo) e a animação e criação de perspectiva em motor de jogos, o *framework* Microsoft XNA. Estas ferramentas serão abordadas com mais detalhes em seções posteriores.

Já no segundo grupo, deve-se obter as características de funcionamento do sistema

a ser modelado e então fazer com que o ambiente virtual reproduza as mesmas ações dado um mesmo estímulo. As características funcionais do sistema são aquelas que descrevem como cada componente se comporta, como por exemplo uma porta ao abrir e fechar. As características lógicas, implementadas na etapa final do desenvolvimento do sistema, dizem respeito às características cujo estudo são a finalidade do experimento em questão. Como exemplo, os autores citam que se o modelo virtual será usado para estudar a ergonomia de uma cabine de avião, as características lógicas são as regras que devem ser programadas para que seja possível estudar a ergonomia.

Para este projeto, os autores tiveram contato com o sistema a ser modelado por aproximadamente quatro meses antes de iniciar o desenvolvimento do sistema de RV. Sendo assim, as funcionalidades do sistema eram conhecidas pelos autores, o que possibilitou a tradução da lógica para o ambiente virtual e a implementação das características funcionais.

Já as características lógicas, que dizem respeito ao funcionamento do sistema para a finalidade de supervisão e treinamento, foram implementadas da forma proposta no procedimento, ao final de grande parte do sistema virtual já pronto. Como exemplo, é possível citar a configuração da movimentação de componentes (características funcionais) para seguir sinais provenientes da comunicação estabelecida com o CLP do sistema real.

3.2 Sistemas supervisórios (SCADA)

No cenário atual da indústria, a automação tem se tornado o epicentro das maiores inovações no setor, e com essa nova realidade alguns interesses surgem, como o conhecimento das variáveis, atuadores e sensores de sua planta automatizada, para melhor conhecimento da planta, dos possíveis estados do processo e seu funcionamento (RODRÍGUEZ et al., 2016). Além dessa crescente, um tema que tem tomado grande importância, seja na indústria como um todo como no âmbito educacional, são os sistemas supervisórios - que tem como principal objetivo auxiliar no controle e prover ao usuário uma melhor visualização dos estados/variáveis do sistema.

Sistemas supervisórios, comumente chamados de SCADA (acrônimo para *Supervisory Control and Data Acquisition*), não são sistemas de controle por completo, mas tem o foco na supervisão do sistema real. Com efeito, sua arquitetura se resume na implementação de um *software* no *hardware* que será monitorado, sendo a interface comumente feita em cima de um CLP ou de uma unidade terminal remota (RTU), assim como um servidor central e operadores terminais (DANEELS; SALTER, 1999).

As principais vantagens que se obtém devido ao uso de sistemas supervisórios são (BARR; FONASH, 2004):

- Transferência de dados em tempo real, possibilitando ao usuário ter conhecimento

do estado atual de suas variáveis e poder atuar na solução de eventuais ocorrências.

- Pré-requisito para a melhora de qualquer passo pertencente à um sistema de produção complexo, corroborando com a importância da noção dos valores dos sensores e atuadores do sistema.

- Diferentes complexidades podem ser modeladas por esses sistemas, bastando apenas a criação de uma IHM que possibilite a interação do sistema com o *hardware* a ser supervisionado.

Nesse aspecto de inovações, nota-se um esforço em diminuir a complexidade no aprendizado destas novas tecnologias - com um enfoque maior em sistemas SCADA - e diversos estudos foram feitos procurando atingir esse objetivo, tais como: desenvolvimento de sistema de treinamento SCADA baseado em computadores pessoais (ABDI et al., 2016), criação de um laboratório SCADA para pesquisa e treinamento (THOMAS; KUMAR; CHANDNA, 2013) e criação de IHM para treinamento do manuseio de sistemas SCADA (ADAMO et al., 2007).

Analisando o sistema de treinamento SCADA baseado em computadores pessoais tratados em (ABDI et al., 2016), o ponto focal do artigo é demonstrar a versatilidade que os sistemas SCADA possuem quando utilizados por computadores pessoais, que viabilizam o acesso dos dados por todas as operações das empresa em questão, bem como seu relativo menor custo e com o aumento relevante do poder computacional, tornando-os atrativos para as aplicações listadas. Desta forma, a construção da IHM para posterior integração com o sistema e seu manuseio, além de didaticamente ser favorecida, é mais viável economicamente quando comparada à compra de um *software* específico para esta atividade.

Já no estudo de caso apresentado por (ADAMO et al., 2007) é ressaltada de fato a importância da IHM em sistemas SCADA, que anteriormente era construída por componentes elétricos ou eletromecânicos, e atualmente passou a ser concebida por interfaces virtuais, que além de propiciar uma experiência mais agradável e didática ao usuário, proporciona um ambiente de fácil manuseio nos processos supervisionados pelo SCADA, seja em sua atuação direta (no controle ou atuação do sistema) como na obtenção dos estados do sistema e de suas variáveis, fatores esses que elucidam as principais vertentes dos sistemas SCADA.

Por fim, no estudo de caso do laboratório SCADA para pesquisa e treinamento é possível fazer um paralelo com os possíveis avanços que podem ser alcançados pelo Laboratório de Sistemas Automatizados presente no Departamento de Engenharia Mecatrônica da Escola Politécnica. No presente estudo é retratado, de maneira didática, como os alunos podem ter um primeiro contato com Sistemas SCADA, no que tange ao

controle, supervisão e gerenciamento dos processos descritos por ele e no entendimento da integração do sistema físico com o meio virtual, tendo sua importância mensurada pelo desenvolvimento de pequenas estações de automação com sistemas de processos distribuídos, suportando assim uma base de dados universal, o que acaba por possibilitar que computadores pessoais possam acessar a rede e acompanhar os diferentes estados que o sistema assume, bem como suas variáveis (THOMAS; KUMAR; CHANDNA, 2013).

Além dos estudos supracitados, um tema de grande relevância nas inovações trazidas por sistemas SCADA é o proposto em (KEBO; STAŠA, 2012). Com o intuito de elucidar essas inovações o estudo previamente mencionado trata da inserção da realidade virtual no controle de processos de mineração, uma vez que diversas atividades podem ser executadas concomitantemente nesta área (como pode ser visto nas Figuras 6 e 7).

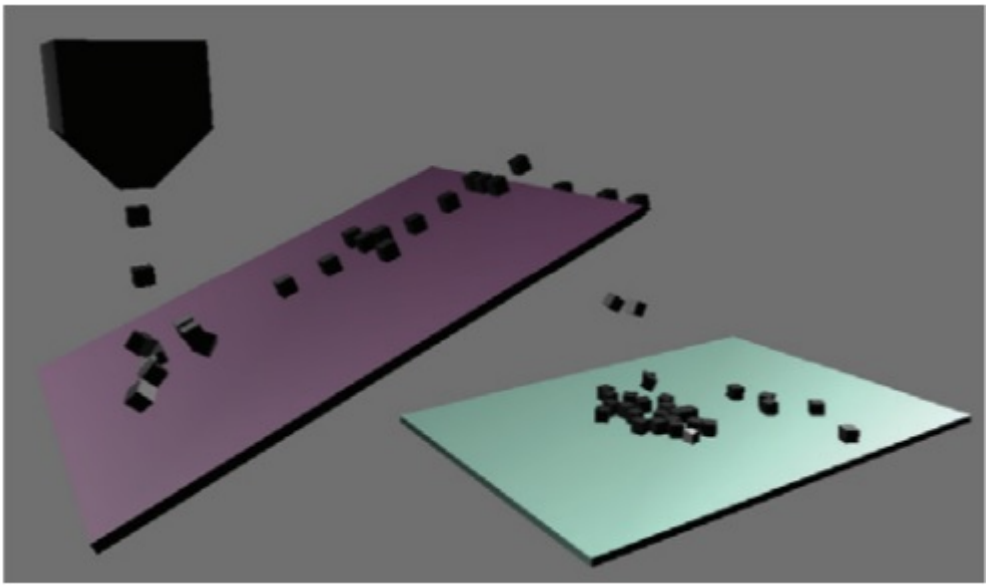

Figura 6 – Exemplo de um modelo virtual utilizado para o transporte de material. Retirado de (KEBO; STAŠA, 2012)

Apesar das diversas diretrizes que o processo pode assumir, ainda existe a necessidade da criação de um ambiente que integre todas as partes do modelo ao mesmo tempo, o que acaba produzindo uma melhor avaliação de como os processos se entrelaçam e que, quando processados em conjunto, possibilitam uma simulação diferente de acordo com os parâmetros intrínsecos escolhidos (tais como paralelismo nas atividades de produção e transporte, robustez das unidades de produção, escalabilidade da quantidade de material utilizado na execução de uma tarefa, dentre outros fatores).

Desta forma, o estudo elenca a importância da realidade virtual em um sistema SCADA tradicional e como essa ferramenta pode ser utilizada como uma aprimoramento no controle dos processos e na criação de novas possibilidades no âmbito de simulação. Com efeito, nota-se a relevância que a RV imprime, possibilitando ao usuário a conexão de métodos de algoritmos inteligentes, viabilizando inúmeros processos para a concepção do

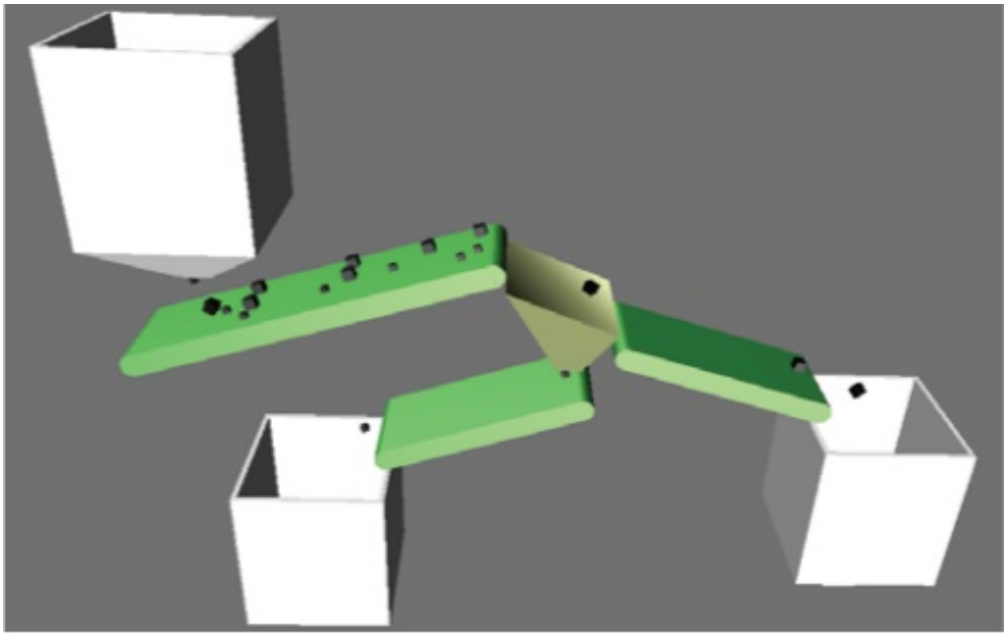

Figura 7 – Exemplo de um modelo virtual de um determinado processo. Retirado de (KEBO; STAŠA, 2012)

processo central em questão, a criação de diferentes ambientes que sejam condizentes com a realidade e, também, o desenvolvimento de novas formas de visualização dos processos modelados, o que acarreta numa melhor avaliação dos resultados da simulação em questões operacionais e estratégicas.

Ainda nessa vertente, é possível fazer a ligação do presente projeto com o estudo realizado no quesito da estruturação da realidade virtual embarcada no sistema SCADA. Ou seja, cria-se diversas possibilidades ao usuário, seja no modo de treinamento, onde atua-se diretamente no processo central da bancada e nos resultados finais, bem como pode-se atuar nas variáveis do estado durante a execução do processo central carregado no CLP, tornando-se possível analisar visualmente as consequências dessas mudanças devido à RV.

3.3 Comunicação via *OLE for Process Control* (OPC)

OPC é um padrão de comunicação usado em aplicações industriais que permite a transmissão de dados entre computadores e máquinas. A comunicação por OPC é independente de fabricante, o que permite a troca de informações entre diferentes componentes sem que ocorram problemas de compatibilidade e sem que seja necessário incluir um *driver* adicional para cada *hardware* na rede. A introdução desse padrão, portanto, reduz a complexidade da implementação de *drivers* para *hardware* para apenas um, o servidor OPC; e a da criação de interfaces em *software* para esses *drivers* também para apenas uma, o cliente OPC (ZHENG; NAKAGAWA, 2002).

Com sua primeira versão lançada em 2006, a manutenção e o desenvolvimento do padrão OPC são realizados pela *OPC Foundation* (Kontron Czech, 2011), que disponibiliza bibliotecas, código-fonte e exemplos práticos para usuários basearem o desenvolvimento de suas próprias aplicações. O próprio OPC é, por sua vez, baseado na tecnologia COM/D-COM (*Distributed Component Object Model* da Microsoft), que estabelece o protocolo de comunicação entre componentes.

As várias especificações que compõe o conjunto do OPC, segundo a *OPC Foundation*, são:

- *OPC Data Access*: define o padrão para troca de dados, incluindo valores de variáveis e tempo de execução;
- *OPC Alarms and Events*: define o padrão para troca de informações do tipo alarme e mensagem, além de gerenciar variáveis de estado; e
- *OPC Historical Data Access*: define métodos de busca e análise de dados históricos coletados e arquivados.

3.3.1 *Distributed Component Object Model* (DCOM)

Devido ao acelerado crescimento da internet em conjunto com o aprimoramento das redes de multimídia e alta velocidade, o processamento distribuído adquiriu um elevado grau de relevância para a implantação de protocolos e arquitetura na infraestrutura das redes atuais. Visando descomplicar o gerenciamento de componentes que são baseados em *software*, criou-se um protocolo de objeto distribuído chamado DCOM (*Distributed Component Object Model*) (KING; HUNT, 2000).

Acerca de suas funcionalidades, o DCOM permite que diversos componentes de um mesmo *software* se comuniquem através de uma mesma rede, garantindo fidelidade e eficiência na troca de informações (BING-HAI; LI-FENG; CHUAN-MENG, 2006). Desta maneira, para que a comunicação entre esses componentes não seja obstaculizada devido à problemas de funcionalidade requerida ou ao uso de alguma linguagem específica, uma arquitetura foi montada (vide Figura 8).

Nesta arquitetura (BING-HAI; LI-FENG; CHUAN-MENG, 2006) estrutura-se dois executáveis COM (*Component Object Model*) - um para o cliente e outro para o servidor) que são responsáveis por prover serviços orientados ao objeto para ambos. Usando um procedimento de chamada remota (RPC - *Remote Procedure Call*) em conjuntura com um provedor de segurança, gera-se pacotes de rede padrão que estão em conformidade com o protocolo DCOM.

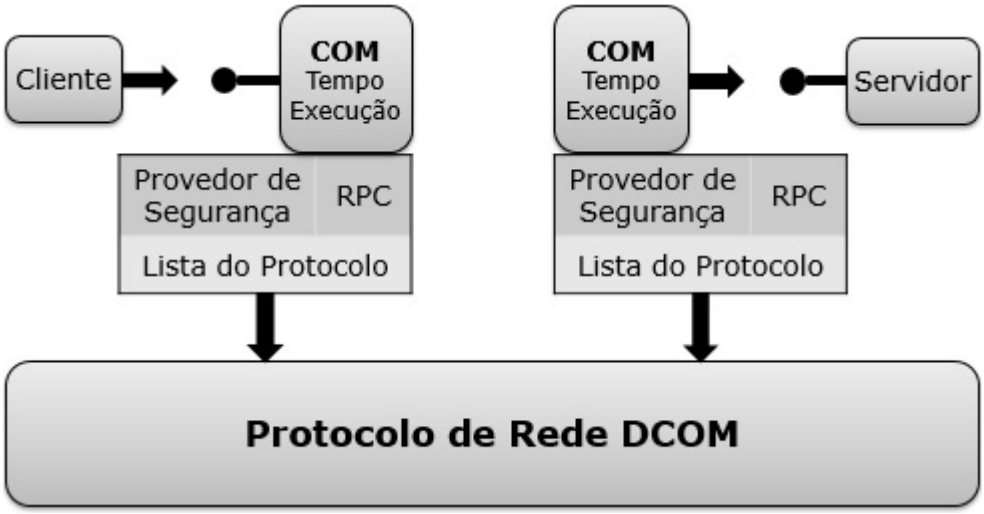

Figura 8 – Estrutura da comunicação segundo o protocolo DCOM. Adaptado de (BINGHAI; LI-FENG; CHUAN-MENG, 2006)

Analisando ainda seu funcionamento, algumas medidas devem ser tomadas para que a conexão entre cliente o e o servidor OPC seja corretamente estabelecida, sendo elas (TAN; YOO; YI, 2009):

1. Autenticações de *firewall* devem ser realizadas com um alto grau de minuciosidade, dado que os maiores problemas de conexão usando este modelo provêm destas.

2. Garantir que a implementação do COM seja confiável caso o sistema utilizado tenha um sistema operacional diferente do *Windows*.

3.3.2 Implantação da Comunicação OPC

Atualmente a comunicação OPC utiliza-se dos serviços de troca de informações fornecidos pelo DCOM (MAHMOUD; SABIH; ELSHAFEI, 2015). Como a pretensão do presente projeto é um sistema supervisório dentro da mesma rede e com poucas limitações impostas pela segurança (no que tange os dados a serem transferidos), o DCOM atende as demandas atuais (OKSANEN; PIIRAINEN; SEILONEN, 2015), sendo baseado numa estrutura de cliente e servidor, conforme pode ser visto na Figura 8.

3.4 Motor de jogos Microsoft XNA

3.4.1 Definição

Segundo (FILHO et al., 2009), Microsoft XNA é um conjunto de bibliotecas de desenvolvimento que tem como intuito otimizar a produtividade de desenvolvedores de jogos no que tange a produção de jogos para Windows ou para o console XBox 360, da própria

Microsoft. Além de encapsular componentes técnicos programados em linguagens de baixo nível, trazendo o foco dos programadores para a experiência do usuário durante o jogo e o seu próprio conteúdo, o XNA possibilita uma rápida curva de aprendizagem (proveniente da simplicidade de manuseio da linguagem C#.NET) e é distribuído gratuitamente pela própria Microsoft.

3.4.2 Lógica de operação

O Microsoft XNA apresenta para o usuário um *framework* de desenvolvimento: um conjunto de soluções para problemas comuns no tipo de aplicação para o qual ele é usado, neste caso o desenvolvimento de jogos.

Um jogo é, do ponto de vista construtivo de *software*, uma aplicação em *loop* infinito que recebe entradas do usuário, calcula novos estados de acordo com a lógica estabelecida para o jogo, e atualiza um interface gráfica que representa visualmente esses estados.O XNA *Framework* tem como um de seus objetivos a resolução dessa dinâmica de maneira a facilitar a construção do jogo pelo programador (BLANCO, 2011).

Nos tutoriais (BLANCO, 2011) e (JUNIOR, 2011), disponibilizados pela Microsoft para familiarização do usuário com o XNA, os autores desenvolvem o conceito de *Game Loop*: a estrutura de implementação da dinâmica comum aos diversos jogos.

Na execução de um jogo, inicialmente existe um processo em que se carrega o conteúdo necessário (e.g. imagens e sons) e ao final esse conteúdo é descartado se necessário. Entre esses dois processos está o *Game Loop*, que se repete continuamente enquanto o jogo não é finalizado e inclui as seguintes etapas:

- ***Update***: executa a lógica de operação do jogo, incluindo a atualização de estados por entrada do usuário e a simulação de efeitos físicos.
- ***Draw***: faz a renderização da visualização e efeitos gráficos.

A sequência de eventos dentro de um jogo desenvolvido com o *framework* Microsoft XNA é resumida na Figura 9.

O *software* desenvolvido neste trabalho tem características semelhantes aos jogos, principalmente no que tange a dinâmica de execução. O sistema supervisório deve a todo momento verificar se houve entrada do usuário ou atualização das variáveis internas do CLP, e a todo momento deve atualizar a visualização para refletir o estado do sistema real.

Por esse motivo, o Microsoft XNA foi selecionado como *framework* de programação. Com o uso desta ferramenta, os autores podem se concentrar na resolução da lógica interna do sistema supervisório executada nas rotinas de atualização, já implementadas no XNA.

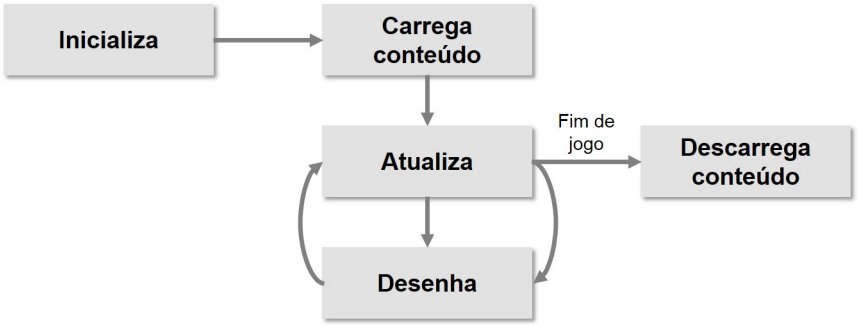

Figura 9 – Sequência de eventos no Microsoft XNA. Adaptado de (JUNIOR, 2011)

3.5 Sistema modular de produção Festo MPS 500

3.5.1 Operação do sistema modular

O sistema modular Festo MPS 500 tem sua operação controlada por um Controlador Lógico Programável (CLP), da mesma forma que muitos processos reais na indústria.

Sabe-se que o CLP tem sido, desde 1969 quando foi introduzido no mercado, a solução encontrada por grande parte das indústrias como plataforma de implementação de algoritmos de controle para as aplicações industriais específicas dos processos de manufatura. Como a gama de produtores desse tipo de controlador era grande, não houve uma padronização no que diz respeito às diferentes sintaxes de linguagem de programação que poderiam ser utilizadas, bem como aos sistemas operacionais e ao ambiente executável da IHM (JAMRO; TRYBUS, 2013).

Com efeito, a Comissão Eletrotécnica Internacional (do inglês, IEC) elaborou uma norma visando padronizar a linguagem de programação dos CLPs, o que permitiu compatibilidade no envio de programas de controle do maquinário independente de seu fabricante.

A norma IEC 61131-3 elenca cinco diferentes linguagens de programação que são efetivamente usadas em CLPs (TISSERANT; BESSARD; SOUSA, 2007):

- Texto Estruturado e Lista de Instruções: por serem linguagens não gráficas permitem ao usuário uma sintaxe de programação mais simples e direta, porém existe a possibilidade do programa ser salvo com falhas semânticas de caracterização do estado, ou seja, o estado não ter sido adequadamente definido e o programa, indiferente a este fato, ser executado erroneamente.

- Diagrama de Blocos de Função: editor gráfico que permite a alocação de elementos de programação - os blocos funcionais - que viabilizam a existência do estado em um *layout* permitido dentro da interface do CLP e que possa ser executado, mesmo com a possibilidade de estar incompleto.

- Diagrama de Funções Sequenciais (SFC): é uma linguagem mais robusta que as textuais pois, além de ser gráfica e mais intuitiva, permite ao usuário a inserção de passos, sejam inicias ou não, assim como transições - convergentes ou divergentes - o que gera um encadeamento de elementos sequenciais, que por sua vez acabam impondo condições de estado do programa.

- Diagrama Ladder: de maneira análoga ao SFC cada *rung* (lógica de controle e programação do Ladder) implica na criação de relés de *output*, ou seja, elementos que caracterizam o estado atual são criados e, por meio da combinação desses, é possível alcançar um novo estado.

Com essas sintaxes de programação, é possível criar Unidades de Organização de Programa (do inglês, POU) (OTTO; HELLMANN, 2009) tais como programas, funções e blocos funcionais. Estes são os principais responsáveis por compor o quadro de tarefas de controle que um CLP deve realizar, desde uma rotina de simples verificação até processos funcionais (OTTO; HELLMANN, 2009).

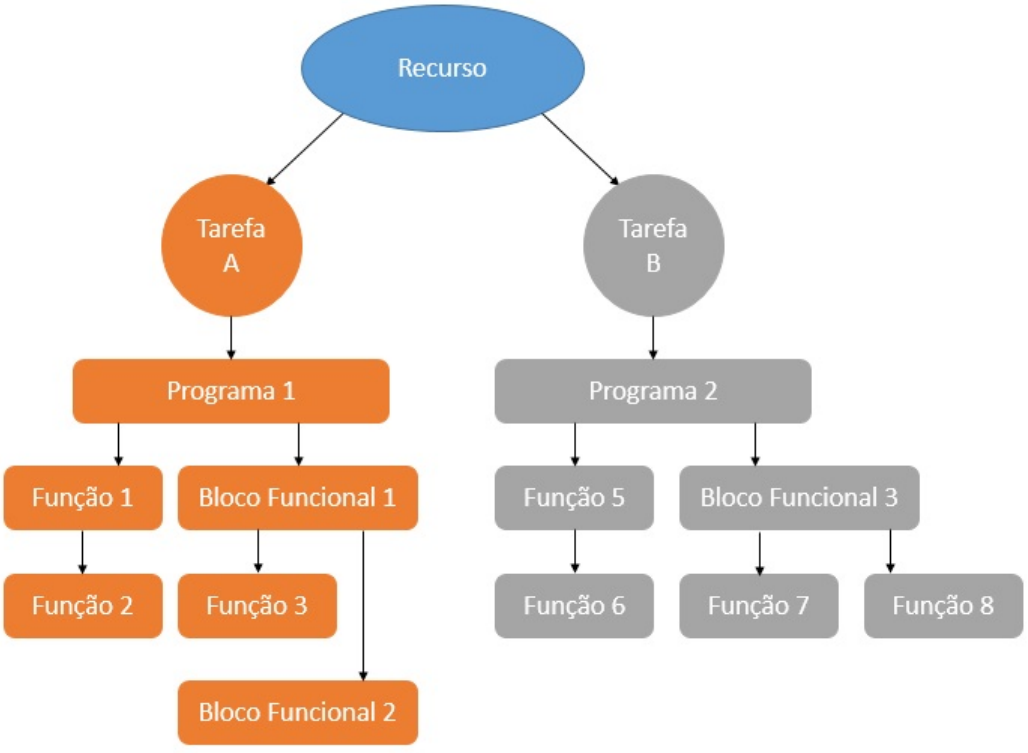

Figura 10 – Estrutura de um CLP segundo a norma IEC 61131-3 e a hierarquia dos elementos internos a essa. Adaptado de (JAMRO; TRYBUS, 2013)

De um modo geral, a norma IEC 61131-3 dita que no sistema de programação de um CLP prevalece uma configuração central, que é composta por diversos recursos, comumente classificados como Unidades Centrais de Processamento (do inglês, CPUs) (OTTO;

HELLMANN, 2009). Sem demasiado esforço computacional, uma ou mais tarefas podem ser executadas em um mesmo recurso, ou seja, esse conjunto de tarefas pode ser executados dentro de uma mesma CPU, ponto esse que apresenta um dos aspectos de relevância destes controladores. Por vezes, a complexidade de uma determinada tarefa pode exigir mais que um programa, e a forma como estes serão executados advém das particularidades da tarefa e dos ciclos de leitura do CLP, que podem variar sua periodicidade.

Um exemplo de estrutura que se enquadra a norma IEC 61131-3 é ilustrado na Figura 10, onde características tais como a junção de tarefas em um único recurso e a hierarquia dos elementos (funções e blocos funcionais) dentro do programa são explicitadas.

4 Definição do Projeto

4.1 Definições

Neste tópico serão apresentadas algumas definições dos termos mais utilizados nesta seção, visando uma melhor compreensão destes. Eles são:

- Sistema Real (SR): sistema modular de produção Festo MPS 500, incluindo tanto os mecanismos de atuação quanto o CLP para controle. Esta é a bancada educacional que simula processos industriais reais (neste caso, um mecanismo seletor) e que será o meio físico sobre o qual o *software* desenvolvido atuará.
- Réplica Tridimensional: modelo CAD da bancada construído no *software SolidWorks* 2016, ou seja, é a representação virtual do sistema real.
- Interface Homem-Máquina (IHM): A IHM é utilizada no intuito de monitorar, controlar, gerenciar e prover visualização gráfica de processos industriais ou de manufatura (YOUNAS; DURRANI; MEHMOOD, 2016), fazendo, assim, com que o usuário tenha acesso ao estado das variáveis e sensores por intermédio de uma interface gráfica em seu computador ou terminal remoto. Ademais, é dada ao usuário a possibilidade de gerir os processos eletromecânicos da planta de produção, conforme pode ser visto no estudo realizado em (YOUNAS; DURRANI; MEHMOOD, 2016), em que os autores propõe a elaboração de uma IHM para controle e gerenciamento de um sistema composto por processo controlado por CLP e monitoramento por sistema SCADA.
- Modelo Virtual (MV): composto pela réplica tridimensional animada com o motor de jogos Microsoft XNA, de modo a capturar a dinâmica do sistema real, e pela IHM do *software*.

4.2 Modos de Operação

O sistema resultante deste projeto possibilita três modalidades de uso. O primeiro modo, ou Modo de Treinamento, é intencionado para usuários que não tem familiaridade com sistemas controlados por CLP ou, mais especificamente, com a bancada Festo MPS. Nesta situação, o usuário tem a possibilidade de interagir com o MV sem a necessidade de se conectar com o SR, ou seja, sem risco de danificar os componentes ou até mesmo de sofrer algum ferimento.

Neste modo, o usuário tem a liberdade de comandar cada um dos atuadores em separado (por exemplo, dar o comando de "ligar a esteira") e verificar a resposta do sistema, sendo que a simulação inclui tanto o comportamento físico (visualização da esteira ligando e peças se locomovendo, caso exista alguma) quanto o comportamento dos sensores (sensor de presença sendo ativado quando uma peça chega em sua zona de alcance). O objetivo deste modo de operação é que um usuário que tem pouca familiaridade com sistemas com CLP tenha uma ferramenta que o auxilie na tradução do processo que ele deseja implementar especificado em forma de linguagem natural para o mesmo processo especificado em forma de programa para CLP, como *ladder* ou SFC.

Como exemplo, caso o usuário queira ativar uma esteira de transporte sempre que uma nova peça entra no sistema e levar todas as peças rosas até uma determinada posição, ele pode entrar com esses comandos manualmente e verificar como as entradas e saídas do CLP se comportam. Sabendo qual é a leitura esperada em cada um dos sensores em um ponto de decisão no sistema (verificação de cor, por exemplo), se torna mais fácil traduzir esse processo para uma linguagem formal de CLP.

Uma limitação de uso deste modo decorre do tratamento de eventos físicos, que foi modelado com complexidade suficiente para que o usuário pudesse ter uma experiência positiva de aprendizagem e interagir com a bancada no intuito de realizar operações dentro de seu escopo de atuação.

No caso da bancada usada neste estudo, o sistema físico é um seletor que tem a capacidade de executar uma lógica de acordo com as características de cada peça introduzida no sistema. A modelagem realizada considera que o usuário pode usar qualquer um dos atuadores, incluindo um sistema de travamento de fluxo, para levar qualquer um dos tipos pré-estabelecidos de peça (três peças cilíndricas, conforme disponíveis no sistema real) para qualquer uma das rampas de saída. Ainda assim, é possível que o usuário realize operações que ocasionem o mau-comportamento do sistema, como por exemplo a deformação e intersecção dos modelos virtuais de diferentes partes do sistema. O efeito visual deste tipo de situação, no entanto, deve ser entendido pelo usuário como uma evidência de que esta operação igualmente não deveria ser executada no sistema real.

No segundo modo, ou Modo de Operação Manual, o usuário ainda tem controle sobre a ativação dos mecanismos, mas agora qualquer ação surte efeito tanto no MV quanto no SR. Desta forma, o usuário tem duas novas possibilidades: I) verificar como o SR se comporta com comandos manuais em uma segunda etapa após o uso do Modo de Treinamento; ou II) realizar testes ao longo da execução de um programa carregado no CLP, com a possibilidade de forçar a atuação de algum componente e verificar o efeito na lógica de execução do programa.

No terceiro modo, ou Modo Supervisório, o usuário pode observar o SR durante a execução do processo coordenado pelo CLP a partir do programa nele carregado e da

evolução das peças dentro do sistema. Com isso, o que se obtém é um sistema supervisório com apoio visual em 3D, de forma a imitar o SR na interface com o usuário. Essa réplica do sistema permite que o usuário possa identificar diretamente em qual ponto de sua planta está ocorrendo uma inconsistência, o que pode diminuir seu tempo de reação e beneficiar a tomada de decisão no monitoramento dos seus processos.

Um resumo dos diferentes modos de uso do sistema é encontrada na Figura 11.

	Treinamento	Operação Manual	Supervisório
Sistemas ativos	MV	MV + SR	MV + SR
Lógica de comando	• MV funciona desconectado do SR • Mecanismos do MV são ativados pelo usuário e funcionamento da planta é simulado, incluindo resposta dos sensores	• CLP controla SR • Atuadores do SR são ativados pelo usuário a partir do MV • Mecanismos são ativados no SR e animados no MV	• CLP controla SR • Sensores do SR são ativados por peça • Mecanismos são ativados no SR e animados no MV
Racional	• Permite que usuários não familiarizados com CLP verifiquem estados do sistema ao longo de determinado processo sem uso do SR	• Permite que o usuário realize testes com o SR agindo diretamente nos atuadores, sem a necessidade de carregar um programa completo no CLP	• Sistema supervisório com apoio visual • Permite a identificação direta de inconsistências a partir do modelo virtual da planta

Nota: MV = Modelo Virtual; SR = Sistema Real

Figura 11 – Descrição das modalidades de uso do sistema

4.3 Arquitetura do Sistema

O sistema desenvolvido é composto por duas partes principais que se comunicam, o modelo virtual (MV) e o sistema real (SR). Partindo do usuário, o fluxo de informações é como se segue:

1. O usuário interage com a IHM do MV e fornece os comandos (seleção de modo, por exemplo);

2. Caso o Modo de Treinamento seja selecionado, o usuário não será conectado ao SR e suas ações serão gerenciadas completamente no MV (incluindo a possibilidade de se adicionar peças virtuais). Para o Modo de Operação Manual, o usuário terá a possibilidade de enviar sinais de comando para os atuadores do SR. Por fim, no Modo Supervisório o usuário não poderá atuar no MV ou no SR.

3. O SR recebe os comandos de atuação a partir do MV ou executa a lógica de controle do CLP e, na medida em que os mecanismos são movimentados, retorna como resposta ao MV o estado de atuadores e sensores;

4. Por fim, a resposta dos sensores e o sinal dos atuadores no SR é usado na animação da réplica tridimensional, que pode ser visualizada pelo usuário por meio da IHM.

A arquitetura do sistema pode ser vista na Figura 12.

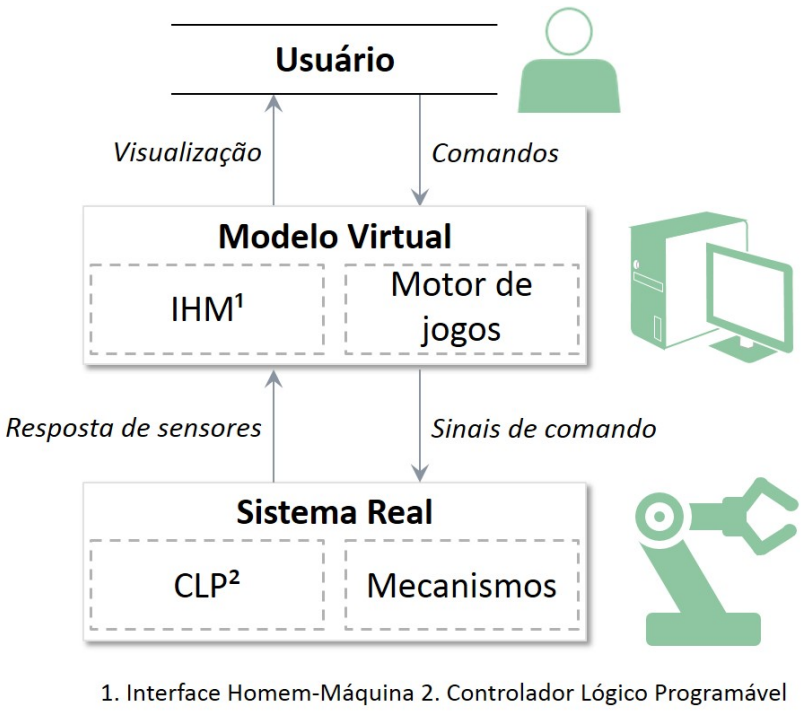

Figura 12 – Arquitetura do sistema

4.4 Requisitos de Projeto

Para ser considerado bem sucedido por todas as partes interessadas, em especial os autores, a banca avaliadora e os eventuais usuários do produto final (e.g. alunos que interagem com as bancadas educacionais, professores e pesquisadores), os requisitos que o sistema deve atender são:

- **Apresentar uma interface gráfica** que permita a seleção dos modos de uso do sistema e que contenha uma réplica tridimensional do sistema real.

- **Permitir a comunicação** entre o sistema real e o modelo virtual, a ser evidenciada pela reprodução dos sinais de sensores e atuadores ativados no sistema real pelo modelo virtual.

- **Reproduzir a operação** do sistema real por meio da animação da réplica tridimensional.
- **Permitir o controle** dos mecanismos do sistema real a partir da IHM, a ser evidenciada pela ativação dos atuadores a partir de comandos na estação em que o *software* é executado.

5 Projeto

5.1 Construção e animação da réplica tridimensional

5.1.1 Modelo tridimensional

Para a construção do modelo tridimensional (réplica tridimensional do sistema modular de produção Festo MPS 500) foi utilizado o *software* SolidWorks 2016 por oferecer uma solução 3D completa para as necessidades do projeto, a saber:

- Possibilitar a elaboração do modelo 3D da bancada com nível de detalhamento o suficiente para a permitir que o usuário reconheça os mecanismos e partes físicas reais no meio virtual;
- Fácil manuseio e possuir ferramentas que auxiliam na construção de protótipos e produtos mecânicos;
- Possuir a extensão de arquivos adequada para o uso na ferramenta de animação escolhida.

É possível ver a bancada modelada na Figura 13.

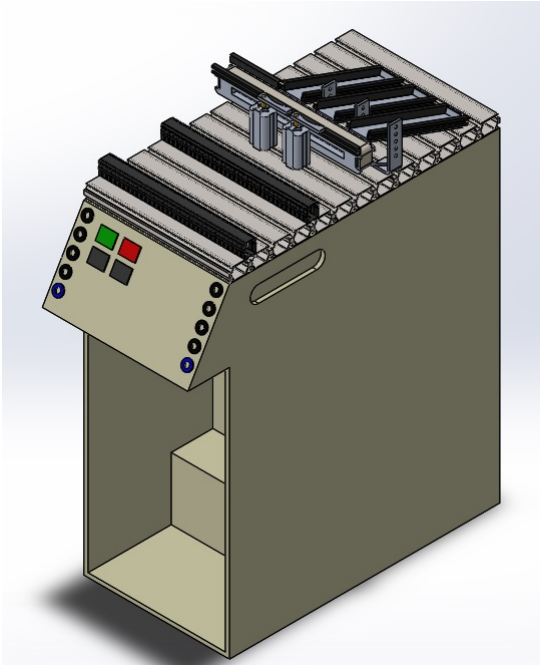

Figura 13 – Réplica tridimensional da bancada realizada no *software* SolidWorks 2016

5.1.2 Preparação do modelo para animação - Autodesk 3ds Max

O 3ds Max é um pacote de ferramentas para confecção de projetos em 3D desenvolvido pela Autodesk Media and Entertainment (AUTODESK, 2016a), sendo altamente empregado na indústria de jogos (tanto de computadores como de console), criação de efeitos especiais para filmes baseados em *scripts* e na criação de propagandas e/ou programas de televisão no ambiente virtual. Tem como principal meta auxiliar na construção de modelos e animações tridimensionais, além da aplicação de texturas, dinâmica de movimento, efeitos visuais e renderização em cima destes.

O *software* foi escolhido devido à interface de integração do modelo animado feito em CAD possuir a mesma extensão dos arquivos de modelo 3D do fabricante (AutoCAD), embora existam outros *softwares* de modelagem que trabalhem com animações, como por exemplo o Blender (BLENDER, 2016). Além disso, para obter melhor desempenho do 3ds Max, o formato de arquivo que será utilizado no presente projeto é o Filmbox (FBX), que também é da Autodesk (AUTODESK, 2016b), formato esse que pode ser exportado para o Microsoft XNA (METLAB, 2016).

5.1.2.1 Recursos e Funcionalidades

Dentre as principais funcionalidades que o software possui, as que mais se destacam para o presente projeto são (DERAKHSHANI; MUNN; MCFARLAND, 2007):

1. Aplicação de texturas – Dentro do *software* existe um editor de texturas que possibilita que uma gama de texturas (vide Figura 14) seja aplicada aos objetos criados e/ou importados ao 3ds Max. Permite desde a combinação de diferentes padrões até a edição das propriedades dos materiais pré-definidos, seja por meio da criação de efeitos (tal como *Hair and fur* – Cabelos e pelos) ou pela mudança de propriedades intrínsecas do material (tal como a escala RGB). No presente projeto a aplicação das texturas foram utilizadas para realçar os detalhes da bancada e retratar a bancada de maneira mais realista ao usuário (como, por exemplo, a distinção nas cores das peças inseridas)

2. Criação de *bones*, *meshes* e *skins* – Por meio do estúdio de personagens é possível criar personagens pela inserção de um esqueleto no objeto modelado, sendo esse esqueleto composto de *bones* (ossos que vão reger a movimentação relativa do objeto), *meshes* (malhas que definem os limites do objeto que será animado, garantindo assim seu dinamismo e realismo – evita-se distorções e a desconstrução do objeto, conforme pode ser visto na Figura 15) e *skins* (que ajustam a deformação da *mesh* quando esta deve ocorrer - movimentos relativos gerenciados pelos *bones*). Como no modelo virtual não existe modelos relativos em um mesma *mesh* não foi necessária a criação

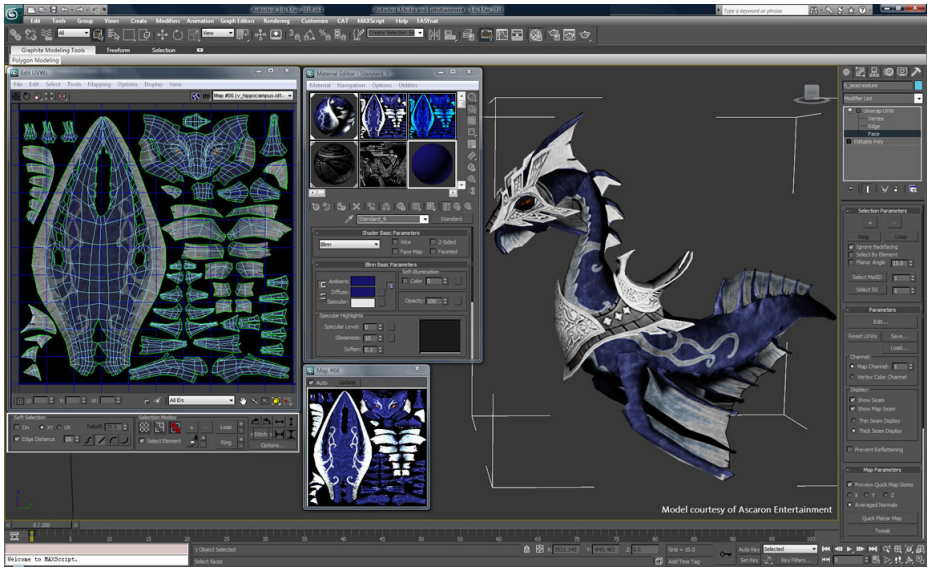

Figura 14 – Editor de Texturas do Autodesk 3ds Max. Retirado de (AUTODESK, 2016a)

de *bones* e *skins*, porém as *meshes* foram essenciais para que os movimentos dos atuadores pudessem ser gerenciados pelo motor de jogos.

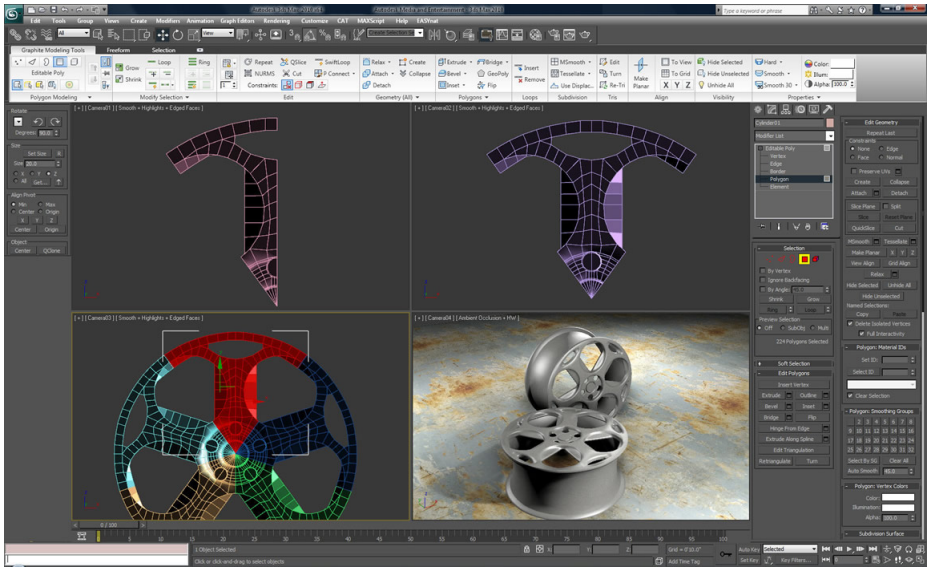

Figura 15 – Utilização das restrições da *mesh* no Autodesk 3ds Max. Retirado de (AUTODESK, 2016a)

3. Facilidade de visualização dos objetos – Por intermédio do explorador de cena é possível navegar com facilidade os diversos elementos que a compõe. O modelo tridimensional é apresentado através de uma lista hierárquica no *software* (enumerada pelas *layers* – camadas) dentro da cena, criando-se uma facilidade na visualização de como os objetos estão organizados, assim como permite-se que os mesmos sejam filtrados ou encontrados em suas respectivas camadas - vide Figura 16. Este aspecto do 3ds Max foi utilizado para melhor hierarquizar as *meshes* do modelo no que tange

sua movimentação, e para que as restrições de posição e localização dos objetos no espaço virtual fossem respeitados.

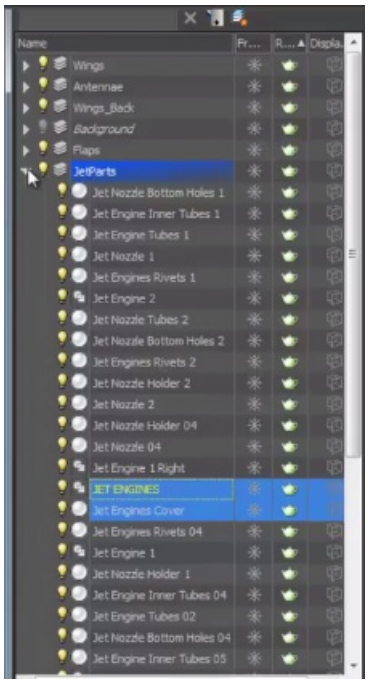

Figura 16 – Organização das *layers* no explorador de cenas. Retirado de (AUTODESK, 2016a)

4. Criação de *scripts* – Visando a otimização do uso da ferramenta, o *software* dispõe de interpretador de *scripts*, o *MAXScript*, quem tem como principal função auxiliar na automação de tarefas repetitivas (tais como ajustes ao modelo CAD importado no propósito de gerar animações) dentro do programa.

5. *Inverse/Forward Kinematics* – Buscando a animação do objeto como um todo através da movimentação de apenas algum membro terminal, o *software* disponibiliza um *solver*. Esta característica do *software* não fora utilizada devido à baixa complexidade de movimentação do MV elaborado, porém é de demasiada importância na concepção de modelos virtuais no próprio ambiente do 3ds Max. Sua consequente importação para o Microsoft XNA é facilitada por intermédio do arquivo FBX, que será tratado na próxima seção.

5.1.2.2 Vantagens de usar o formato Filmbox (FBX)

Sendo originalmente desenvolvido pela Kaydara, o FBX é de propriedade da Autodesk e é amplamente utilizado por oferecer as seguintes vantagens (AUTODESK, 2016b):

- Intercambialidade entre *softwares* – Por ser classificado como um arquivo que facilita a troca de dados entre ferramentas, possibilita o envio de modelos 3D com alto grau de fidelidade.

- *Workflow* aprimorado e interoperabilidade – Por possuir SDK's (*Software Development Kits* – Kits de Desenvolvimento de *Software*) eleva-se a facilidade de operação em diversas ferramentas. Ademais, aprimora-se o fluxo de trabalho pelo fato do arquivo vir conjuntamente com um conjunto de ferramentas proprietárias desenvolvidas especificamente à ele e comumente utilizadas em produções de filmes, desenvolvimentos de jogos e na área de propaganda e *marketing* digital.

5.1.2.3 Modelo CAD animado pelo 3ds Max

Utilizando-se das características do Autodesk 3ds Max supracitadas na seção 5.1.2.1 foi obtido o modelo ilustrado na Figura 17, respeitando-se as restrições de posição e características intrínsecas de material.

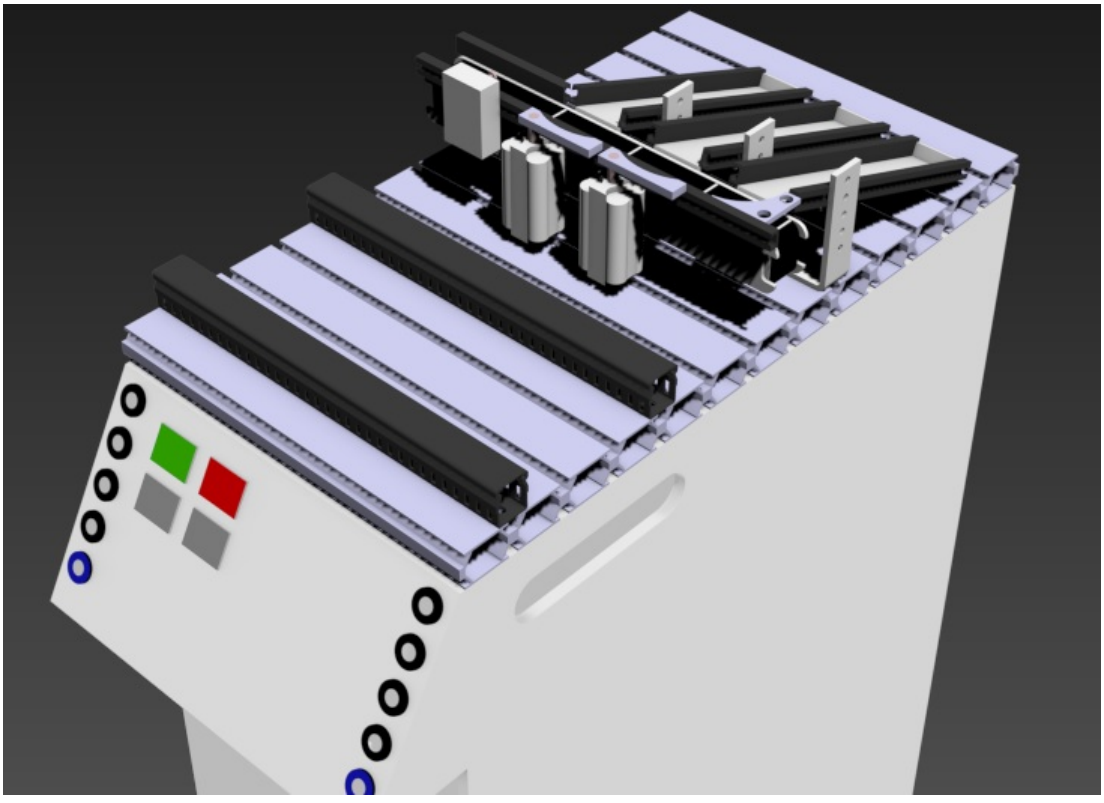

Figura 17 – Bancada modelada através da importação do CAD e aplicação das principais funcionalidades do *software* Autodesk 3ds Max

5.2 Diagramas UML

A UML (*Unified Modelling Language*, do inglês) é uma linguagem de modelagem visual que tem como principal propósito determinar, visualizar, elaborar e documentar os principais elementos e características que um sistema de *software* possui (RUMBAUGH; JACOBSON; BOOCH, 1999). Com efeito, nota-se a relevância da utilização desta metodologia no presente projeto pois ela permite a abstração em alto nível dos requisitos do *software* a ser concebido. Além disso, o UML permite uma comunicação mais eficiente entre a arquitetura e as funcionalidades do sistema (DZIDEK; ARISHOLM; BRIAND, 2008), garantindo que todos os requisitos levantados em fase de concepção de projeto estejam refletidos no *software* resultante.

Os diagramas UML que serão utilizados são:

- Casos de Uso: Descrição das funcionalidades do programa através das possíveis ações que o usuário pode exercer no sistema;
- Atividades: Detalhamento das atividades pertinentes de cada caso de uso, definindo-se assim as operações necessárias para sua implantação;
- Classes: Captura dos principais aspectos da estrutura do sistema, além de uma abordagem inicial sobre a visão geral da programação;
- Sequenciais: Demonstração da interação entre as classes e visualização sequencial do sistema em suas principais atividades.

5.2.0.1 Diagrama de Caso de Uso

O primeiro diagrama a ser construído é o de casos de uso, que descreve as funcionalidades do programa por intermédio das ações do usuário. Deste modo o diagrama é representado na Figura 18 e o detalhamento das atividades é descrito a seguir.

- Controlar os atuadores no Modo Teste e Operação Manual – Com o olhar sobre o sistema SCADA criado, nestes modos de uso o usuário terá a possibilidade de influir diretamente nos *inputs* dos atuadores do sistema, ou seja, ao alterar os valores de entrada dos atuadores, poderá visualizar tanto no MV (no caso de Modo Teste) quanto no SR + MV (no caso do Modo de Operação Manual), como o sistema se comporta e quais mudanças os atuadores causam no processo. Ademais, será possível corrigir inconformidades a partir da alteração dos valores supracitados.
- Escolher Modo desejado/Retornar ao Menu Principal – Como base do programa, neste caso de uso o usuário seleciona o passo que deseja realizar dentre os três possíveis (vide Figura 11), podendo assim usufruir das suas principais funcionalidades e verificar o cumprimento dos objetivos elencados no escopo do projeto.

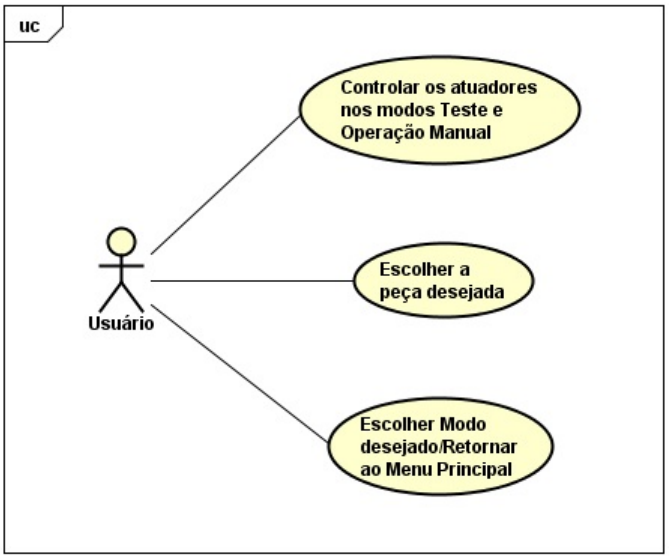

Figura 18 – Diagrama de Caso de Uso do usuário para o sistema

- Escolher a peça desejada – Aqui dar-se-á a opção usuário de escolher a peça que ele deseja inserir na bancada,*input* este importante para o sequenciamento de operações no processo executado por ela no modo teste. No caso dos modos de operação manual e supervisório a cor será obtida através da leitura dos sensores, conforme fora descrito na explicação da Figura 11.

5.2.0.2 Diagramas de Atividades

Nesta seção serão tratados os diagramas de atividade, que detalham as atividades pertinentes de cada caso de uso apresentado na seção anterior, além de como o sistema interage com o usuário no decorrer da sua execução.

Na Figura 19 é demonstrado o diagrama de atividade responsável pela dinâmica dos modos de teste e operação manual do *software* proposto no que tange a atuação direta no valor dos atuadores. Neste caso, o usuário poderá realizar as seguintes operações:

- Retornar ao Menu Principal - Em qualquer momento da execução do modo o usuário poderá cancelar suas operações e voltar ao menu principal, seja após alterar o valor de alguma variável, apertar o botão referente à esta ação ou ao fechar a caixa de diálogo de ambos modos.

- Alterar Valor dos Atuadores - Visando o principal foco deste modo, será dada ao usuário a possibilidade de alterar os valores de entrada dos atuadores pelo MV, sendo estes automaticamente enviados ao SR (no caso do modo de operação manual). Desta forma, o usuário poderá ver tanto o funcionamento do programa inserido no CLP, compreendendo assim a maneira que os sensores atuam no programa no caso

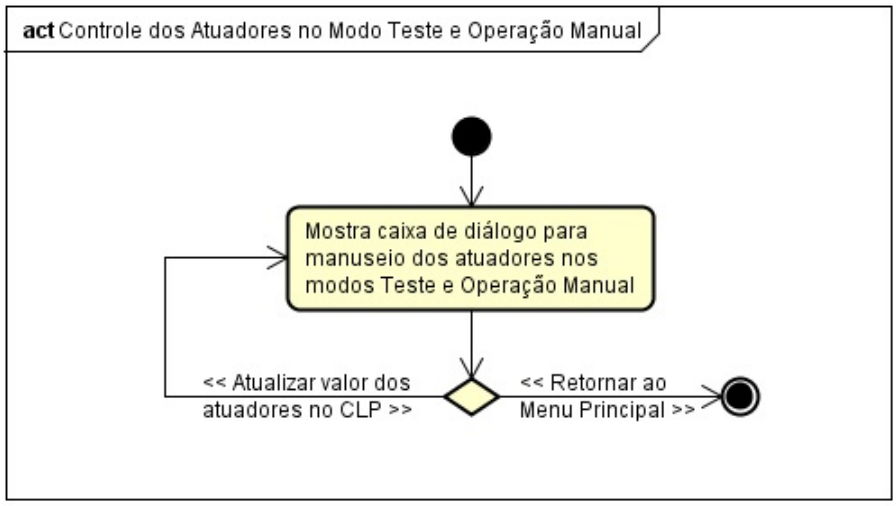

Figura 19 – Diagrama de Atividade - Controle dos Atuadores nos Modos de Teste e Operação Manual

do modo de operação manual, quanto as possibilidade de funcionamento do MV no caso do modo teste.

Na página inicial do *software*, o usuário terá como interface botões para os modos desejados (Teste, Operação Manual e Supervisório - SCADA). Uma vez selecionado o modo, a tela do modo desejado será mostrada ao usuário. Este caso de uso é ilustrado pela Figura 20.

O último caso de uso estruturado ao usuário é o mostrado na Figura 21. Neste caso em específico, o usuário terá a possibilidade de escolher a peça que deseja inserir no modo teste. Vale ressaltar que nos demais modos haverá a detecção da cor da peça pela leitura dos sensores e o usuário deixará de ter a opção de escolher qual a cor da peça que deseja visualizar no MV.

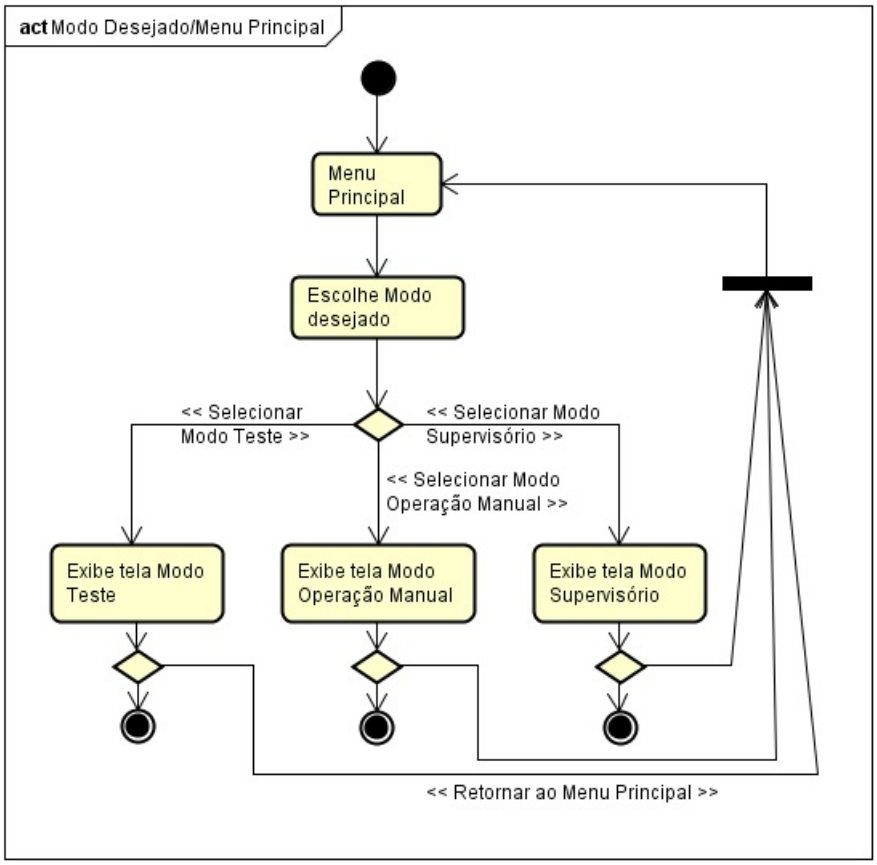

Figura 20 – Diagrama de Atividade - Escolha do Modo Desejado/Retornar ao Menu Principal

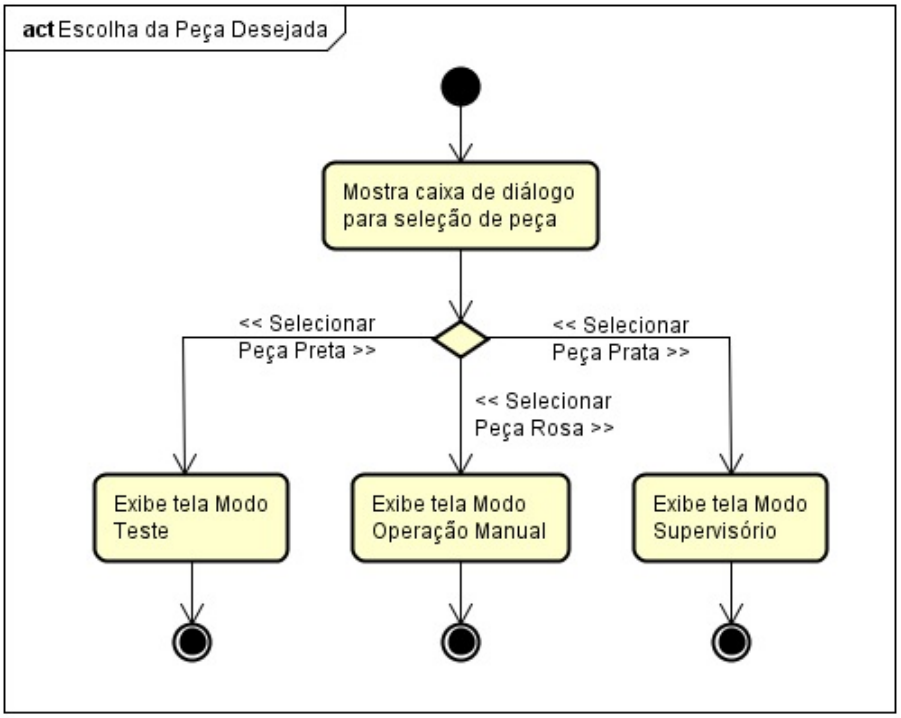

Figura 21 – Diagrama de Atividade - Controle dos Atuadores no MS

5.2.0.3 Diagramas de Classes

Uma vez que os diagramas de atividades foram realizados, criou-se uma primeira versão do diagrama de classes, que tem como principal objetivo elucidar as estruturas que constituem o *software* em questão e que por intermédio de suas interações cumprem com os requisitos elencados nos casos de uso. Conforme a implementação do projeto seja executada, as interações ilustradas na Figura 22 devem ser mantidas e respeitadas.

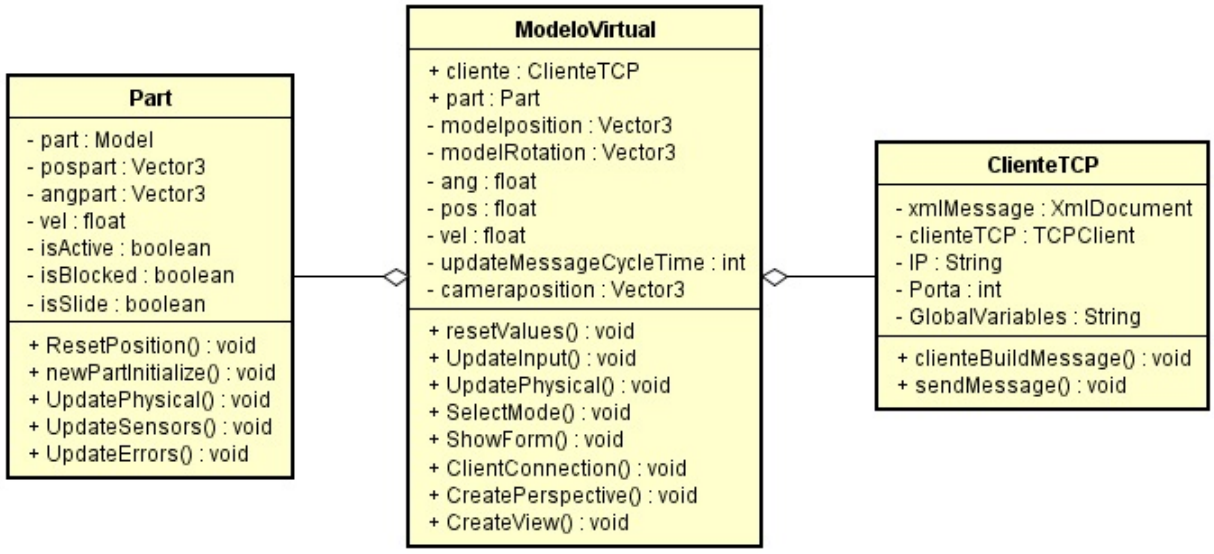

Figura 22 – Diagrama de Classes

A classe **ModeloVirtual** é responsável pelo gerenciamento das principais funcionalidades do *software* proposto no presente projeto. Ela implementa a seleção dos modos de execução (teste, supervisório e operação manual) através da função **SelectModel**, ajusta tanto a perspectiva de visualização quanto a própria exibição do modelo no motor de jogos por intermédio das funções **CreatePerspective** e **CreateView** e estrutura a IHM de interação do MV com o uso da função **ShowForm**. Além destas tarefas, ela efetua um *reset* dos valores da bancada para início das operações em quaisquer um dos modos existentes pela função **resetValues** e atualiza tanto as variáveis de entrada e o estado do MV através das funções **UpdateInput** e **UpdatePhysical**, respectivamente. Por fim, ao chamar a função **ClientConnection** ela inicia a conexão TCP/IP via *socket* TCP do servidor com o cliente.

Por outro lado a classe **Part** tem como intuito modelar e inicializar as peças no modelo, manufatura essa que é a principal causa do fluxo observado durante todo o processo da bancada. Além de dar um *reset* na posição da peça para a posição inicial na bancada pela função **ResetPosition**, ele cria a instância que será exportada para o modelo virtual através da função **newPartInitialize** e assim como fora feito na bancada, atualiza a sua posição através da função **UpdatePhysical** e para o acompanhamento

do estado do MV e SR ele atualiza os valores do sensores conforme a peça se desloca na bancada pela função **UpdateSensors**. O tratamento dos possíveis erros durante a execução do programa é realizado pela função **UpdateErrors**.

Já na classe **ClienteTCP** toda a arquitetura de tratamento de dados do CLP (tanto no envio quanto no recebimento via XML) foi estruturada. Através da função **clienteBuildMessage** o arquivo XML para envio de dados ao servidor TCP, onde é necessário estabelecer-se uma nova conexão TCP/IP, é criado. Logo em seguida, a função **sendMessage** envia o *stream* de dados do arquivo XML transformado em bytes, funcionalidade esta importante para a leitura da mensagem recebida pelo servidor, convertendo-a assim para valores de *input* das variáveis do CLP.

5.2.0.4 Diagramas Sequenciais

Visando agora o detalhamento da interação entre o usuário e os diferentes objetos criados no diagrama de classes, diagramas de sequência foram construídos para representar a funcionalidade de cada caso de uso.

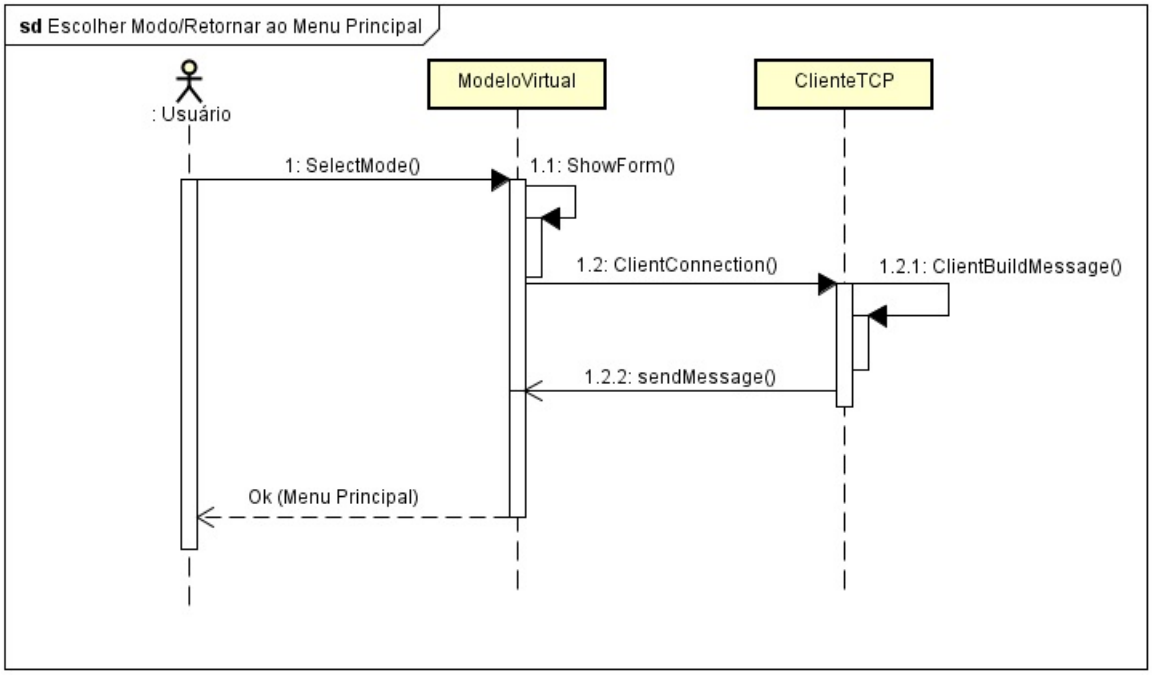

Figura 23 – Diagrama de Sequência - Escolha de Modo/Retorno ao Menu Principal

No diagrama ilustrado pela Figura 23 apresenta-se a forma com a qual a interação entre usuário e *software* ocorre para que o caso de uso de seleção de modo funcione. Como primeiro o passo o usuário deve selecionar o modo que gostaria de utilizar, e isso lhe será oferecido pela classe **ModeloVirtual** pela função **ShowForm** exibida logo após a requisição provinda do usuário através da função **SelectMode**. Uma vez que o modo tenha sido selecionado, a classe **ClienteTCP** é chamada para que a conexão entre

cliente e servidor seja obtida, construindo-se assim um canal no qual mensagens serão enviadas ao servidor (**ClientBuildMessage**) e mensagens oriundas dele serão recebidas (**sendMessage**). Ao término desta sequência, o usuário tem duas opções: a de continuar no modo atual ou de sair dele, retornando assim ao menu principal do sistema.

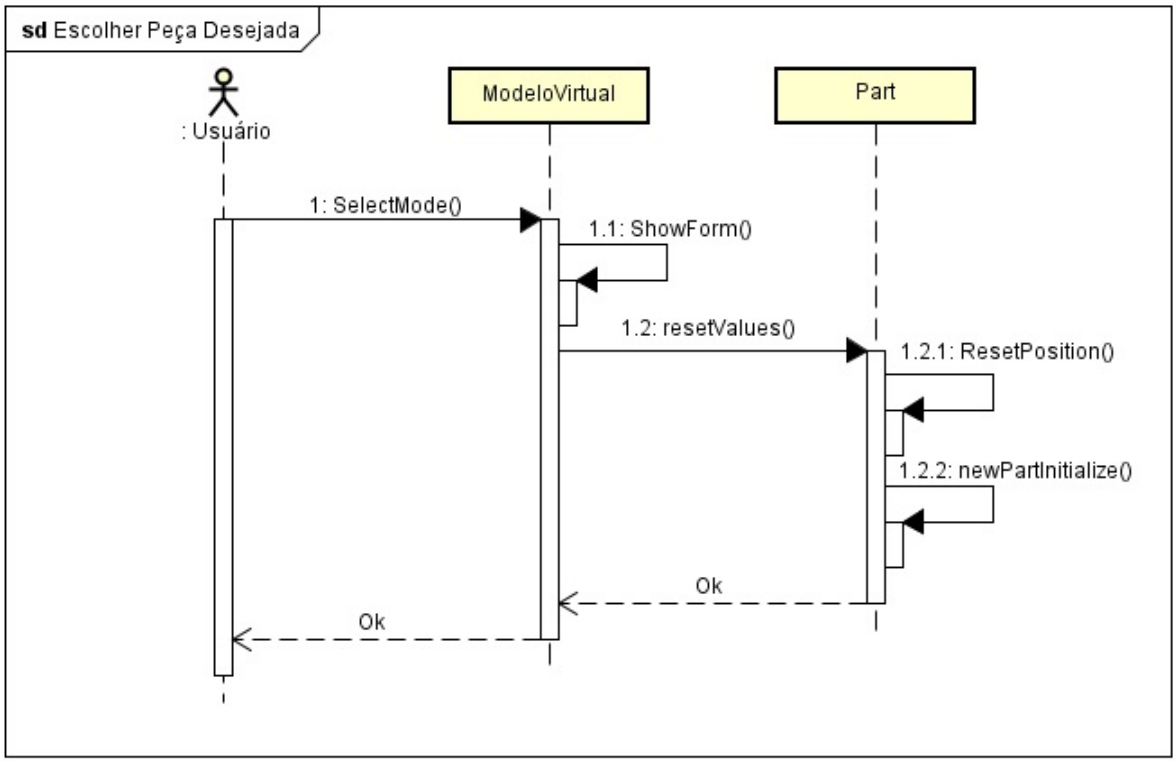

Figura 24 – Diagrama de Sequência - Seleção de Peça

Já no diagrama da Figura 24 nota-se a sequência de comandos que o sistema executa para que a seleção de peça torne-se disponível ao usuário. Da mesma forma que no diagrama anterior, uma vez que o usuário selecione um modo, se este possuir a opção de selecionar uma peça, a função **ShowForm** lhe dará a possibilidade de selecionar uma peça dentre as três possíveis. Uma vez escolhida, a classe **Part** é chamada para que a posição relativa da peça no modelo seja reinicializada e uma nova instância seja criada conforme a preferência do usuário.

Por fim, no diagrama esquematizado pela Figura 25, toda a parte de controle dos atuadores, realizados tanto no modo de treinamento como no modo de operação manual, é estruturada. Aqui, assim como fora feito na seleção de modo, é necessária a escolha do modo para que a conexão com o servidor seja estabelecida e, caso o modo possua a opção de atuação no sistema, essa sequência seja inicializada.

Através da atuação direta do usuário por intermédio do **ModeloVirtual**, a função **UpdateInput** atualiza os valores das variáveis do CLP no que tange os atuadores, e quando esses valores são alterados é necessária a verificação tanto de possíveis erros quanto da posição correta da peça no sistema. Com efeito, as funções **UpdatePhysical**,

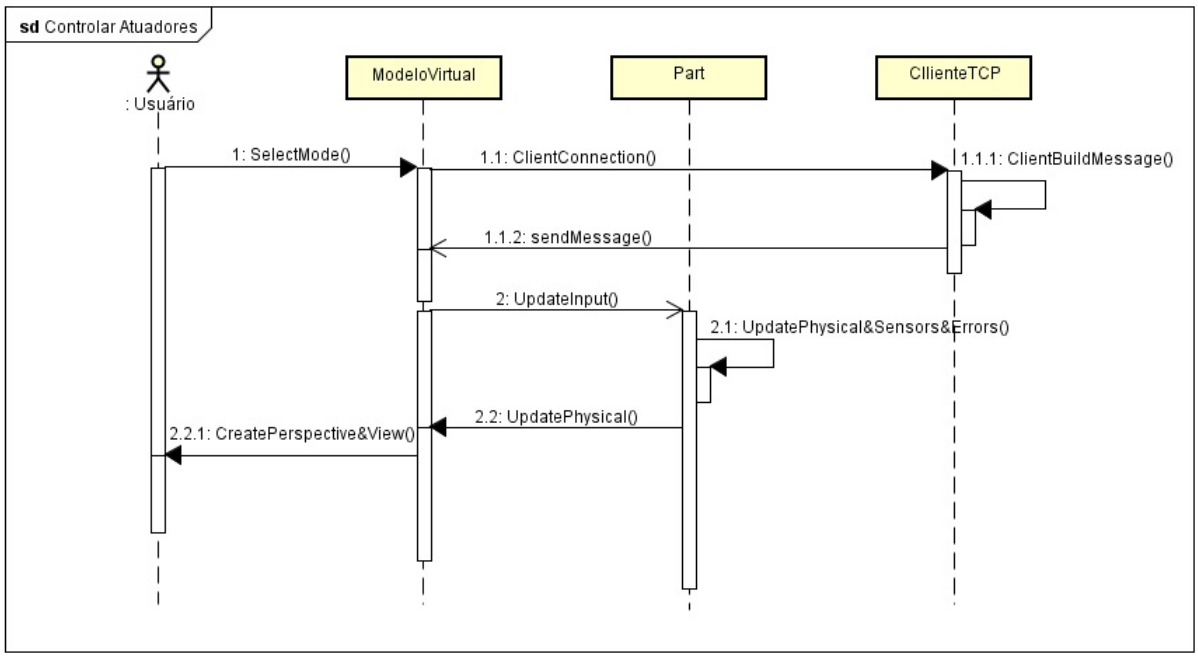

Figura 25 – Diagrama de Sequência - Controle dos Atuadores

UpdateSensors e **UpdateErrors**, presentes na classe **Part**, são responsáveis por realizar essa tarefa e, assim, retornar à classe solicitante a função **UpdatePhysical**, que atualizará as posições de todos os demais componentes do modelo virtual. Adicionalmente, o sistema renderizará todo o modelo virtual para a visualização do usuário por meio das funções **UpdatePerspective** e **UpdateView**.

6 Implementação

6.1 Biblioteca de conexão OpcRcw.Da.Dll

O primeiro passo no desenvolvimento do sistema aqui proposto foi o estabelecimento da conexão entre o computador e a bancada educacional. O padrão de comunicação OPC foi usado com essa finalidade, já que o seu uso em ambas as pontas da comunicação garante que os sinais trocados entre os componentes serão reconhecidos um pelo outro.

A conexão é estabelecida em duas etapas: I) entre o servidor OPC e o CLP; e II) entre o servidor OPC e o cliente. O servidor OPC usado neste estudo é o fornecido pela empresa *CODESYS*, o *CODESYS OPC Server*. Este servidor é fornecido como parte integrante do pacote de desenvolvimento de código em IEC 61131-3 da *Codesys* e sua principal função é a troca de dados entre o computador e o controlador (CODESYS,). Além disso, o servidor deve ser acessado pelo cliente, que é a aplicação que usa os dados da comunicação OPC para uma finalidade própria. No presente estudo, o *software* desenvolvido é um cliente OPC que usa os sinais recebidos do servidor na apresentação de uma interface gráfica para o usuário.

Para estabelecer essa conexão, a aplicação cliente (ou seja, o *software* aqui desenvolvido) deve ser capaz de acessar o endereço do servidor na rede. A solução usada para esta finalidade foi a biblioteca OpcRcw.Da.Dll, que oferece um método de conexão com o servidor a partir de seu nome. Como exemplo, o nome do servidor aqui usado é *FestoCoDeSys.OPC.02*.

Com a conexão estabelecida, é possível buscar as variáveis globais do CLP, que são criadas na etapa de desenvolvimento do seu programa de controle. Essas variáveis são atreladas a endereços específicos de entrada e saída do CLP e têm uma *tag* de identificação, usada quando se acessa o estado da variável por meio da comunicação OPC.

Com o objetivo de validar a etapa da comunicação, foram criadas *tags* para cada um dos atuadores do mecanismo da bancada educacional e para o seu conjunto de sensores. Cada uma dessas variáveis foi, então, lida e alterada com sucesso por um programa de testes, mostrado na Figura 26.

Quando acoplada ao código desenvolvido com o motor de jogos Microsoft XNA, essa mesma biblioteca possibilita que se estabeleça a conexão com o CLP e que sejam alterados o estado dos atuadores da bancada. Sendo assim, esta solução atende à necessidade do projeto de estabelecer uma conexão entre cliente e servidor OPC dentro de uma plataforma que possibilita a animação de um modelo tridimensional.

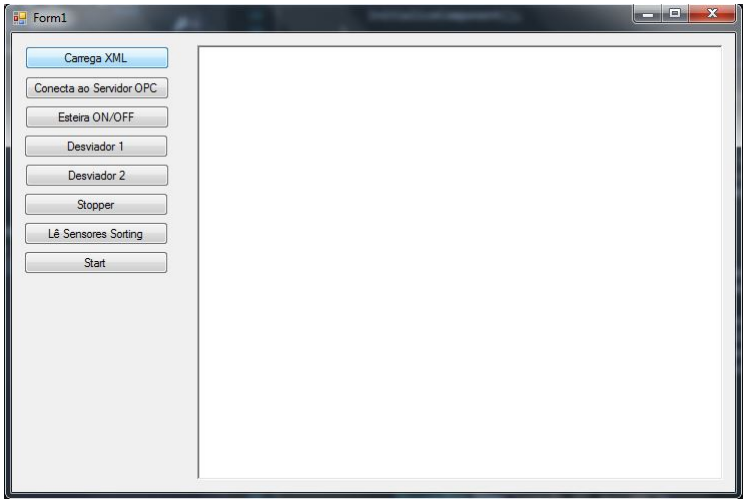

Figura 26 – Interface do programa usado para teste de conexão. Botão "Carrega XML" faz o carregamento das configurações de servidor e *tags* do OPC, mantidos isolados do código fonte para facilitar sua alteração.

6.1.1 Estabelecimento de comunicação por *socket* TCP (*Transmission Control Protocol*)

Uma capacidade adicional para o projeto é a possibilidade de acesso remoto a partir de conexão via internet, em especial para uso remoto do sistema em modo supervisório de uma planta produtiva. Neste caso, o usuário poderia, por exemplo, ter um centro de controle único para plantas em diferentes locações. Para tornar o sistema preparado para este tipo de acesso, foi necessário implementar mais uma camada de cliente e servidor, desta vez executando um *socket* de comunicação por protocolo TCP.

Para esta etapa da comunicação, foi necessário incorporar uma ferramenta que permitisse a transferência de uma sequência de *bytes* ordenados contendo qualquer comando entrado pelo usuário e que pudesse ser capturado no lado do servidor TCP. A partir disso, esses *bytes* poderiam ser tratados pelo cliente OPC, que traduz a entrada do usuário para comandos que devem ser enviados ao CLP por meio da comunicação OPC.

O *socket* desenvolvido foi composto por duas partes:

- **Servidor TCP**: estabelece um *listener* que aguarda alguma chamada no endereço de IP e porta especificados. Assim que chega uma nova requisição de comunicação, captura o *stream* de dados na linha de comunicação e envia uma resposta. Este lado do *socket* roda em uma CPU conectada à bancada (na mesma aplicação do cliente OPC, já que os dados capturados são então processados por este). O método implementa um *loop* infinito, que força o servidor a permanecer aguardando uma nova mensagem na porta especificada. Para que isso não causasse o congelamento da interface com o usuário, o método é lançado a partir de uma *thread* em segundo

plano no programa principal.

- **Cliente TCP**: inicia a conversação com chamadas para o servidor e envia os dados entrados pelo usuário. Este lado do *socket* é remoto e pode rodar em qualquer máquina que tenha acesso ao IP do servidor - seja por rede local ou por internet. O método é gerenciado por uma classe específica chamada **ClienteTCP** na aplicação principal deste projeto (que contém o modelo virtual da bancada).

Para possibilitar o tratamento sistemático da mensagem transmitida entre cliente e servidor no *socket*, foi utilizada a linguagem de marcação XML (*eXtensible Markup Language*). Sendo assim, toda mensagem enviada pelo ciente e toda resposta do servidor na contém uma mensagem no formato XML que pode ser percorrida pelo lado oposto e que traz todas os valores necessários para o acionamento e leitura dos componentes da bancada educacional.

6.1.2 Implementação do código principal

O código principal da aplicação que é o produto final deste projeto foi desenvolvido usando a linguagem de programação C#, o *framework* para criação de jogos Microsoft XNA 4.0 e o ambiente de desenvolvimento Microsoft Visual Studio 2013.

Conforme detalhado anteriormente, o Microsoft XNA estabelece um ciclo de execução que passa pela atualização de variáveis, tanto de entrada do usuário quanto internas do programa, e pelo desenho de itens gráficos. Os dois principais componentes que implementam a rotina estabelecida pelo XNA são as classes **ModeloVirtual** e **Part**. A primeira faz a gestão da bancada e seus mecanismos, além de concentrar o desenho de todos os itens gráficos do programa. A segunda faz a gestão das peças que são processadas na bancada, atualizando seu estado e variáveis de posição.

Para a interface de entrada de comandos do usuário, foram implementados formulários do tipo *Windows Forms*. Esses formulários gerenciam todas as entradas do sistema, tanto a seleção de modos de uso quanto os comandos de mecanismos.

Nas seções a seguir, a implementação de cada uma dessas partes será discutida em termos de sua estrutura e funcionamento. Para maiores detalhes, o código fonte é anexo a este texto e pode ser usado como referência.

6.1.2.1 Gestão dos mecanismos da bancada

A gestão da bancada é executada na classe **ModeloVirtual**. Além disso, essa classe implementa funções básicas como a configuração de janela de visualização e o carregamento de outros componentes do programa (como a classe de tratamento de peças).

Os métodos de atualização dessa classe, usadas na etapa de *Update* do ciclo do XNA são o **UpdateInputs** e o **UpdatePhysical**.

O primeiro inicializa os formulários de entrada do usuário e implementa a lógica de alternância entre eles. O objetivo deste método é que o usuário sempre tenha uma interface de entrada e que esta seja a correta para a operação que ele deseja realizar.

O segundo contém a lógica de atualização das variáveis de posição de cada uma das partes da bancada no modelo virtual. Este método avalia se I) o usuário entrou com algum comando de atuação ou II) o CLP enviou algum sinal de estado ativado por meio da comunicação OPC e atualiza a variável de posição condizente para a parte que está ativada. A título de ilustração, o trecho de código abaixo executa a atualização da variável que guarda o ângulo de rotação de um dos desviadores de peças da bancada caso este componente esteja ativado (e decrementa essa variável no caso contrário).

```
    // Gerencia DESVIADOR 1 ------------
if (GlobalVariables.desviador1Value != "0")
    if (angDesv1 < (float)(Math.PI /4))
        angDesv1 += velDesv;
if (GlobalVariables.desviador1Value == "0")
    if (angDesv1 > 0.0f)
        angDesv1 -= velDesv;
```

As variáveis de posição atualizadas no método **UpdatePhysical** são, então, usadas na etapa de *Draw* do ciclo do XNA. Nesta etapa, o modelo da bancada desenvolvido no 3DS Max é percorrido em todas as suas *meshes* e cada uma delas é desenhada de acordo com a sua referência em relação ao resto do modelo e com a posição instantânea calculada e armazenada nas variáveis de posição.

Para fazer o desenho da *mesh* na posição correta, o método **Draw** implementa transformações lineares de translação e rotação de acordo com os vetores de posição de cada parte da bancada. Inicialmente, a **mesh** é transportada para a origem do sistema de coordenadas global, o que facilita a movimentação em torno de um eixo local da peça. Após a atualização da posição local, a peça é novamente transportada para sua posição em relação ao outros componentes da bancada, usando a informação que é guardada na hierarquia de posições do modelo.

6.1.2.2 Gestão de peças

A bancada modelada tem o objetivo de avaliar peças de acordo com critérios físicos (cor e material) e executar uma lógica de seleção. A gestão da réplica virtual dessas peças é realizada no programa pela classe **Part**, que é um componente de jogo do tipo XNA.

Essa classe faz a inicialização de uma nova peça no modelo virtual de acordo com a entrada do usuário ou com os sinais do CLP. No Modo de Treinamento, o usuário tem a liberdade de incluir uma nova peça e deve especificar qual é o tipo desejado dentre os três disponíveis (peça preta, rosa ou prata). Já no Modo Supervisório, a peça é inicializada assim que o sensor de presença disponível na bancada é ativado. Inicialmente, essa peça não tem um tipo específico e é representada como um cilindro em branco. Assim que os sensores de avaliação de cor e material identificam o tipo da peça, o modelo virtual é atualizado para representar a peça que está sendo processada.

Da mesma forma que na classe **ModeloVirtual**, a classe **Part** implementa o método **UpdatePhysical**, que atualiza as variáveis de posição da peça presente no sistema. Para executar a lógica de atualização de posição, o estado dos atuadores da bancada é avaliado para que se verifique se a peça deveria estar em movimento. Como exemplo, se a peça está sobre a esteira e ela é ligada, a peça ganha velocidade e se desloca. Caso algum dos desviadores ou o bloqueador de fluxo sejam ativados, a posição da peça é avaliada e, caso ela esteja dentro da região de conflito, sua trajetória é alterada de maneira similar ao que ocorre no sistema real.

6.1.2.3 Gestão de entradas do usuário

A entrada do usuário é obtida por meio de janelas do tipo formulário. A todo instante existe uma janela ativa para que usuário possa selecionar o modo de uso, entrar com comandos ou visualizar o estado de atuadores e sensores do sistema.

Inicialmente, o usuário é apresentado a um menu de seleção de modo, que pode ser visto na Figura 27.

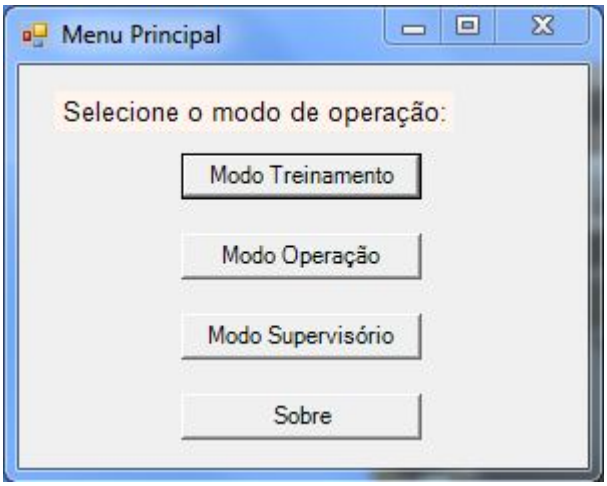

Figura 27 – Menu de seleção de modo

Após a seleção de modo, é aberta uma janela de controle pertinente ao modo selecionado. A configuração destes formulários é muito similar, sendo que a maior diferença

está na liberdade que o usuário tem para entrar com comandos no sistema. Nos modos Treinamento e Operação Manual, o usuário pode comandar atuadores individualmente, enquanto que no Modo Supervisório a interface é restrita à apresentação do estado da planta e o usuário não tem liberdade para dar comandos.

O exemplo do formulário exibido no Modo de Treinamento é mostrado na Figura 28. Nela, é possível ver as duas seções de itens: **Atuadores** e **Sensores**. O estado de cada um destes é representado por um conjunto de marcadores gráficos e seu valor é exibido em uma caixa de texto. Por fim, o endereço de cada entrada ou saída do CLP é fornecido para facilitar o uso das informações no momento de programação do CLP.

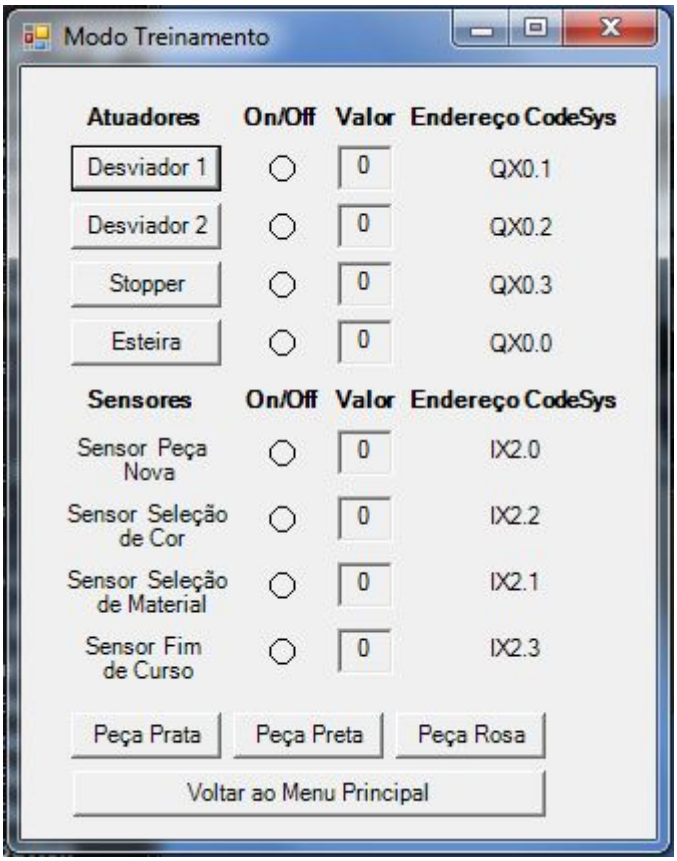

Figura 28 – Interface com usuário no Modo de Treinamento

7 Resultados

O resultado final do programa desenvolvido, sendo executado no Modo de Treinamento pode ser visto na Figura 29. A seguir, o resultado será avaliado nos diferentes modos de uso do *software*.

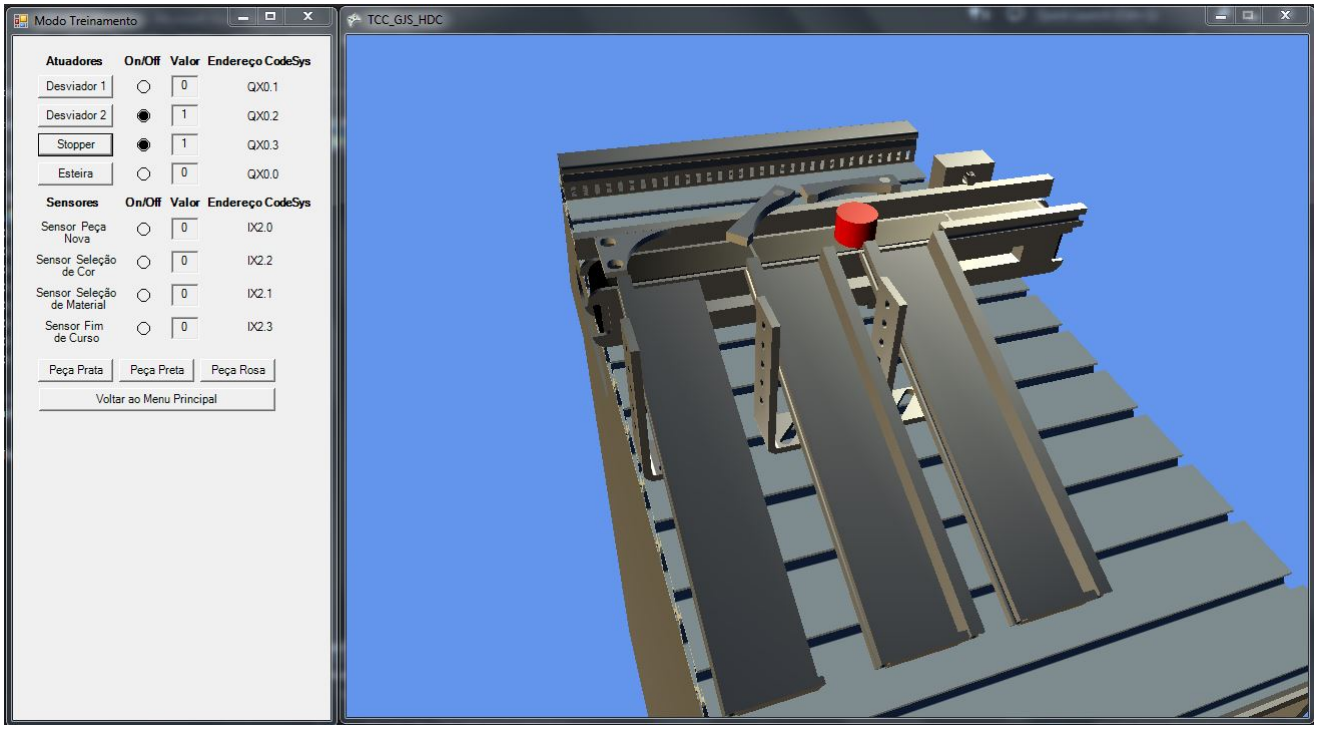

Figura 29 – Exemplo de visualização do programa. Neste caso, o Modo treinamento está sendo executado e existe uma peça virtual no sistema.

7.1 Modo de Treinamento

Para avaliar a funcionalidade do *software* no Modo Treinamento, foram realizados testes de comparação dos resultados simulados com o uso do modelo virtual com os resultados esperados de acordo com a operação normal da bancada educacional.

Os testes tiveram dois objetivos principais: I) verificar aderência da animação do modelo virtual ao comportamento do componentes físicos da bancada e II) verificar a resposta dos sensores na IHM com o processamento de uma peça virtual.

Para o primeiro objetivo, o *software* foi usado junto da bancada, ainda que desconectado, e foram verificados tanto o tempo dispendido para cada uma das ações quanto a comparação visual da movimentação das peças. Já para o segundo, cada um dos sensores foi testado com peças virtuais para avaliação da resposta obtida. A resposta foi comparada

com a esperada de acordo com a alocação de endereços de saída do programa carregado no CLP e os resultados obtidos foram positivos.

Na Figura 30 é mostrado um exemplo desses testes. Na imagem à esquerda, uma nova peça acaba de ser adicionada e, desta forma, aciona o sensor de chegada de peça na bancada. Logo após, a esteira é acionada e a peça se desloca até o sensor de avaliação de material, que retorna a resposta esperada para o tipo de peça selecionada.

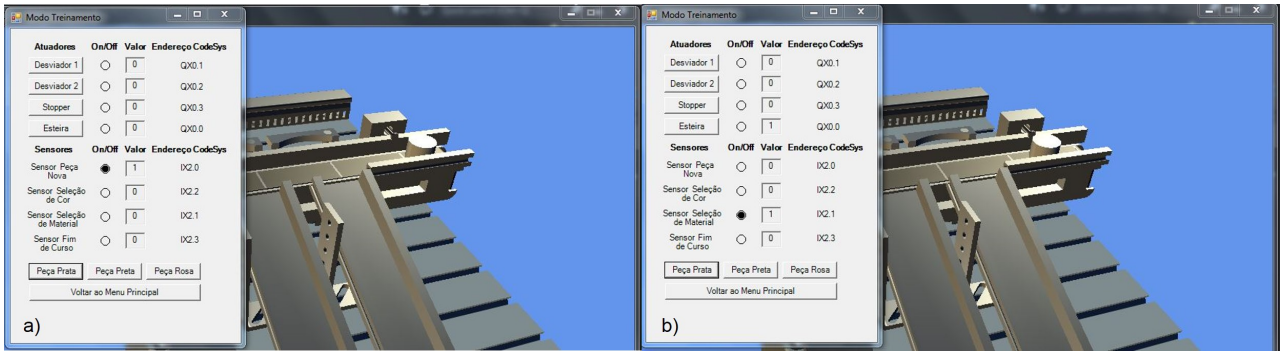

Figura 30 – Teste de resposta dos sensores no Modo de Treinamento: a) sensor de chegada de nova peça é ativada pela entrada de peça virtual; e b) sensor de material detecta peça metálica.

Com os testes realizados, verificou-se que o modelo virtual se comporta de maneira similar ao sistema real sem a necessidade de ativar os mecanismos da bancada. Desta forma, é possível que um usuário não familiarizado com este tipo de sistema se utilize deste modo de operação para um primeiro ensaio livre de riscos com a bancada. Além disso, mesmo um usuário que tenha algum grau de familiaridade poderia usar este modo para testar a resposta dos componentes do sistema sem precisar, por exemplo, realizar uma conexão com o CLP.

7.2 Modos de Operação Manual e Supervisório

Para avaliar a funcionalidade do *software* nos Modos Operação Manual e Supervisório, foram realizados testes de comparação da animação do modelo virtual com a operação simultânea do sistema real.

Os testes tiveram dois objetivos principais: I) verificar aderência da animação do modelo virtual ao comportamento do componentes físicos da bancada e II) verificar a resposta do sistema supervisório em caso de erro.

Para o primeiro objetivo, o *software* foi usado em conjunto com a bancada e com o estabelecimento da comunicação entre a CPU executando o programa final e o CLP da bancada. Foram, então, realizadas tomadas de tempo para diferentes ações (deslocamento de peça do início da esteira até o primeiro desviador, por exemplo), inicialmente com o

sistema real e depois com a animação do modelo virtual. Foi constatado que a animação segue o sistema de forma que o erro é imperceptível ao usuário, de forma que a reprodução pode ser usada na supervisão da planta.

Adicionalmente, um teste realizado foi o de reconhecimento de peça, que garante que a animação seja condizente com o objeto introduzido no sistema. Os sensores disponíveis para essa identificação são: sensor de presença, que indica se existe uma nova peça no início da esteira independente de seu tipo; sensor de cor, que indica se a peça é escura ou clara; e sensor de material, que indica se a peça é metálica ou não. Ao se colocar uma nova peça na esteira, o sensor de presença é ativado e o modelo virtual apresenta uma peça branca, já que o tipo ainda não é definido. No momento em que a peça chega à região de identificação de cor e material, o modelo virtual é capaz de alterar a representação da peça para o tipo correto. Em todos os testes realizados, a identificação se deu da forma esperada.

Por fim, para o segundo objetivo, foram identificados dois erros potenciais que seriam tratados pelo sistema. Eles são:

- **Remoção de peça** por agente externo, seja por operador humano ou por colisão com outro objeto ao longo do trajeto; e
- **Limite de peças atingido** em cada uma das rampas de saída, sendo que o limite é considerado como o máximo de peças que pode ser armazenado em uma rampa sem que ocorra falha no processo por ativação incorreta de sensores.

Para o primeiro erro, o comportamento do modelo virtual é comparado com os sinais de saída do sistema real. Como o modelo virtual está limitado aos sensores instalados na planta, entre um sensor e outro ele irá assumir que a operação está se desenvolvendo conforme o esperado, ou seja, sem a remoção indevida da peça. No caso em que o modelo virtual atinge um estado de avaliação de sensor e o sistema real indica algo fora do esperado, entende-se que houve extravio da peça e o usuário é alertado. O sistema com mensagem de erro é mostrado na Figura 31, em que a peça havia sido removida pelo operador para efeito de teste.

No tratamento de erro de rampa cheia, o usuário é alertado assim que qualquer uma das três rampas recebe o total de seis peças, independente de quais sejam. Com o alerta, o usuário deve esvaziar a rampa e indicar ao sistema, o que reinicia a avaliação de preenchimento para aquela rampa. A mensagem de erro com indicação da rampa a ser esvaziada pode ser vista na Figura 32.

Desta forma, foi avaliado que o sistema é capaz de identificar alguns dos principais erros que podem ocorrer na operação do sistema real. Aqui, as possibilidades de tratamento são limitadas pela capacidade de sensoriamento da bancada educacional. Como não é

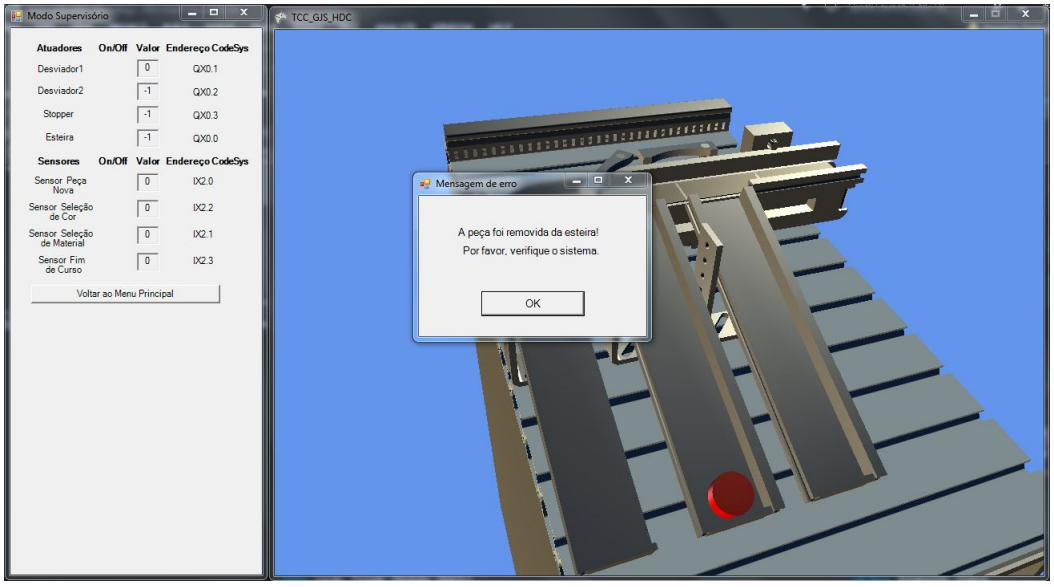

Figura 31 – Indicação de erro no caso de peça removida do sistema. Neste caso, a peça foi propositalmente removida da esteira pelo operador para efeito de teste.

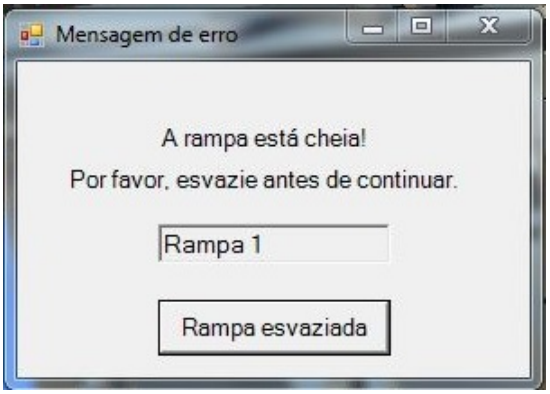

Figura 32 – Indicação de erro no caso de rampa cheia. A mensagem especifica a rampa a ser esvaziada e dá a opção de reiniciar a avaliação.

possível saber com certeza a posição da peça a todo instante, a avaliação deve ser feita em locações específicas, o que pode gerar atraso entre a ocorrência do erro e sua identificação.

Uma das dificuldades na implementação destes modos de operação foi o estabelecimento de comunicação com a bancada educacional de forma a ler e escrever valores nas variáveis globais do CLP. O resultado com a configuração de comunicação implementada foi obtido com êxito dado que permitiu:

- **Leitura e escrita de variáveis no CLP**, evidenciadas em testes de acionamento de atuadores e de captura de resposta dos sensores à excitação simulada. Os dados capturados foram comparados com a resposta esperada dos sensores, calculada de acordo com a alocação de bits nos endereços de saída do CLP.
- **Acesso remoto** do computador que executa o programa com o modelo virtual ao

computador que roda o servidor TCP e se comunica com a bancada. Com isso, é possível manter um equipamento dedicado à comunicação com os diferentes CLPs de uma planta e acessar os dados capturados neste por meio de outro computador que tenha acesso ao endereço de IP do servidor.

- **Atualização imediata de estados de acordo com entrada do usuário**, ou seja, tempo de comunicação imperceptível ou ao menos não prejudicial à experiência do usuário. Mais especificamente, o ciclo de atualização de mensagens entre servidor e cliente da comunicação por *socket* TCP é de 50 milissegundos, o que garantiu que nenhum atraso fosse percebido na operação do sistema ao longo dos testes.

- **Manutenção de integridade da mensagem XML**, evidenciada em teste de avaliação da mensagem trocada entre cliente e servidor do *socket* TCP. A Figura 33 contém a interface utilizada para a visualização das mensagens XML recebidas pelo servidor. Ao longo dos testes, não foram observados erros de comunicação ou mensagens parciais, cortadas na transmissão.

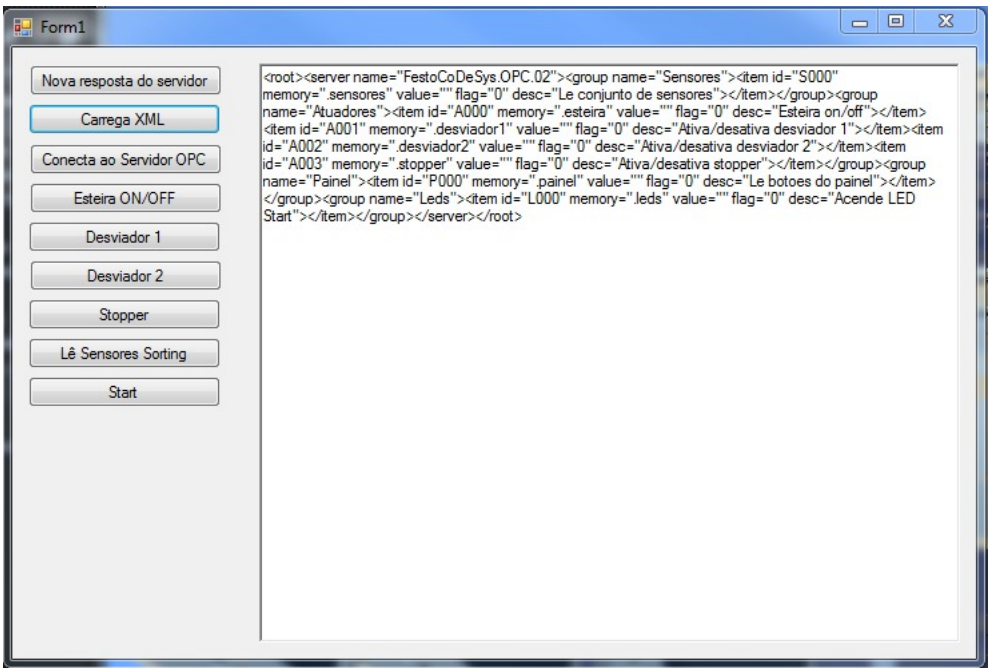

Figura 33 – Interface utilizada nos testes de integridade da mensagem XML.

7.3 Aderência aos requisitos de projeto

Visando avaliar o sucesso do projeto, é necessário que haja um olhar crítico sobre o cumprimento, ou não, dos requisitos de projeto elencados anteriormente pelos autores. Desta forma, é possível discriminar os requisitos de projeto e analisar a aderência do sistema final com eles. Logo:

1. **Apresentar uma interface gráfica**: No presente requisito esperava-se que o sistema disponibilizasse ao usuário a possibilidade de seleção de modos por intermédio de uma IHM e que houvesse uma réplica tridimensional que auxiliasse na construção do sistema requerido. Ambos aspectos foram cumpridos, pois além do *software* possuir a réplica tridimensional concebida no sistema virtual, criou-se uma ambiente ao usuário no qual ele consegue navegar entre diferentes modos (teste, operação manual e supervisório) e usufruir, de maneira didática e simples, das principais funcionalidades do sistema.

2. **Permitir a comunicação**: No que tange à comunicação espera-se que o SV fosse capaz de se comunicar com o SR, fato este comprovado através da supervisão do sistema e atuação direta no processo através do modo de operação manual.

3. **Reproduzir a operação**: Conforme citado acima, é nítida a aderência do sistema ao requisito proposto pois o modo supervisório (implementação do sistema SCADA na bancada) foi concluída com êxito, como pode ser visto da seção de resultados.

4. **Permitir o controle**: De forma análoga à reprodução da operação, permitiu-se ao usuário o controle sobre o processo da planta no modo de operação manual, no qual o usuário tem liberdade de se sobrepor à lógica de operação presente no CLP e intervir nos atuadores.

Portanto, nota-se que o sistema obtido pelos autores reproduziu, de maneira efetiva, todos os requisitos de projetos inicialmente propostos, e abre lacunas para futuro progressos nesse âmbito, elencados na seção de Futuras melhorias do capítulo de Conclusão.

7.4 Replicabilidade dos resultados

Os resultados deste projeto foram obtidos a partir de um sistema específico, o módulo seletor da bancada educacional Festo MPS 500. No entanto, da mesma forma com que existem particularidades de implementação do projeto que estão atreladas ao uso desta bancada, existem conceitos e ferramentas que podem ser usados na replicação dos resultados para outros sistemas.

Nesta categoria, o ponto fundamental é a ideia de que o processo de construção aplicado neste estudo, que foi detalhado ao longo das seções anteriores, pode ser replicado para outras plantas que sigam algumas características básicas, dentre elas:

- Planta controlada por CLP.
- Acesso à variáveis globais de programação do CLP, usadas na interface de comunicação OPC.

- Disponibilidade de uma rede local cabeada para comunicação com o CLP e CPU para execução ininterrupta dos servidores TCP e OPC.

Existem alguns fatores que agem em favor da replicabilidade do projeto, dentre eles:

- Universalidade do protocolo de comunicação OPC, que pode ser usado de forma independente do fabricante de qualquer equipamento usado.
- Possibilidade de interação com mais de um CLP simultaneamente, o que confere escalabilidade ao projeto. Esta característica, ainda que não tenha sido implementada no presente estudo, é viável por conta da capacidade do protocolo OPC de lidar com a comunicação com múltiplos CLPs, sendo que o esforço de implementação de código para isso não seria elevado.
- Possibilidade de conexão via internet, o que reduz a limitação física de onde o sistema pode ser instalado e abre a possibilidade de centralização do controle e operação de diversas plantas.

No entanto, existem também fatores que dificultam a replicação do projeto. Dentre eles, os mais relevantes são:

- Necessidade de modelagem da planta, tanto de características gráficas quanto de comportamento físico. Este fator é especialmente importante quando se considera o aumento na complexidade da planta (e.g. número de atuadores, tipos de sensores, etc)
- Esforço de implementação de código, já que uma planta com características diferentes poderia requisitar um tratamento particular na lógica do código da aplicação.

8 Conclusão

O presente trabalho expôs o processo de criação de um sistema para supervisão e treinamento em planta controlada por CLP com uso de conceitos de Realidade Virtual. O processo usado aqui pode ser replicado, ainda que com esforço adicional de modelagem e implementação, para outros ambientes de interesse, contanto que apresentem algumas características básicas como ilustradas na seção de Replicabilidade dos resultados.

Apresentou-se o desenvolvimento deste sistema em todas as suas etapas, incluindo projeto e implementação. São descritos a modelagem da bancada em SolidWorks e 3DS Max; os diagramas UML do *software*; a arquitetura de comunicação entre o programa e o CLP da planta; e a implementação do código principal que executa as funcionalidades desejadas, com uso do *framework* Microsoft XNA.

O resultado final, obtido em forma de *software*, foi de grande êxito por atender aos requisitos de projeto e apresentar novas funcionalidades ao sistema em que foi implementado, uma bancada educacional em laboratório da Escola Politécnica da USP. Os autores esperam que esse resultado possa ser usado por alunos da Escola ao longo dos próximos anos, melhorando a experiência de aprendizagem e pesquisa, e que ele seja replicado e aprimorado por outros que tenham interesse em fazê-lo.

8.0.1 Futuras melhorias

Embora apresente características que complementam a operação de uma planta tradicional, existem pontos de melhoria que poderiam ser introduzidos como refinamento ou como funcionalidades adicionais no *software* desenvolvido.

Algumas dessas potenciais melhorias são discriminadas a seguir:

- **Adaptação para sistema de Realidade Virtual totalmente imersivo:** com o intuito de tornar a experiência do usuário ainda mais interessante e realística, um passo importante seria o de flexibilizar a interação com o sistema usando dispositivos como óculos de visão tridimensional e luvas com sensoriamento. Desta forma, seria criada uma experiência imersiva do usuário no sistema, facilitando a sensação de presença.
- **Refinamento da modelagem de eventos físicos:** a aplicação de diferentes técnicas para o tratamento de colisão entre a representação virtual das várias partes do sistema real, em especial as peças, poderia ser um próximo refinamento do programa. O uso de técnicas avançadas nesse campo poderia trazer novas aplicações para o *software*, como por exemplo o uso em fase de prototipagem de novas peças para que

se avaliasse o resultado da interação dessas com a planta ainda no plano virtual, reduzindo o gasto com protótipos físicos.

- **Discriminação de peça por identificação única:** no intuito de seguir tendências da indústria, um caminho para que se aumentasse tanto a flexibilidade do processo produtivo quanto da detecção de erros é a avaliação de peça com identificação única, ou seja, a capacidade de o *software* entender não só características genéricas como cor e material da peça, mas sim qual é exatamente a unidade que está sendo processada e qual é o destino que ela deve seguir. Essa avaliação pode ser feita, por exemplo, com a introdução de identificação por radiofrequência.

- **Melhoria do sensoriamento do sistema real:** considerando o sistema com base no qual o *software* foi implementado, a bancada educacional Festo MPS, uma medida que poderia auxiliar na fidelidade do Modo Supervisório e na detecção de erros é a inclusão de novos sensores ao longo do trajeto das peças. Como o modelo virtual é limitado aos sinais que recebe do CLP, ele só entende que houve um erro nas posições em que a peça deveria estar passando por um sensor e, portanto, pode apresentar atrasos na detecção de erros.

- **Integração de outros sistemas reais no modelo virtual:** incorporar outros sistemas pode trazer diversas vantagens. Em especial, destaca-se: a supervisão de todo o trajeto da peça ao longo de sua vida dentro do sistema de processamento; e a possibilidade de testar a integração de diferentes sistemas no espaço virtual, antecedendo, por exemplo, a instalação dos sistemas reais.

Referências

ABB. Symphony® Plus. 2012. Disponível em: <http://new.abb.com/power-generation/symphony-plus/simulator>. Citado 2 vezes nas páginas 6 e 12.

ABDI, S. U. et al. Development of PC-Based SCADA Training System. 2016. Citado na página 20.

ADAMO, F. et al. SCADA/HMI systems in advanced educational courses. *IEEE Transactions on Instrumentation and Measurement*, v. 56, n. 1, p. 4–10, 2007. ISSN 00189456. Citado na página 20.

AUTODESK. *3ds Max | 3D Modeling and Rendering Software | Autodesk*. 2016. Disponível em: <http://www.autodesk.com/products/autodesk-3ds-max/overview>. Acesso em: 14 de set. 2016. Citado 4 vezes nas páginas 6, 35, 36 e 37.

AUTODESK. *FBX - Overview*. 2016. Disponível em: <http://www.autodesk.com/products/fbx/overview>. Acesso em: 14 de set. 2016. Citado 2 vezes nas páginas 35 e 37.

BARR, D.; FONASH, P. M. Supervisory Control and Data Acquisition (SCADA) Systems October 2004. *Technical Information Bulletin 04-1*, n. October, p. 76, 2004. Citado na página 19.

BING-HAI, Z.; LI-FENG, X.; CHUAN-MENG, Y. DCOM and MMS-based control software architecture for automated manufacturing system. *International Journal of Advanced Manufacturing Technology*, v. 27, n. 9-10, p. 951–959, 2006. ISSN 02683768. Citado 3 vezes nas páginas 6, 23 e 24.

BLANCO, D. Introdução ao Game Loop com XNA Game Studio. v. 699716, p. 3–7, 2011. Citado na página 25.

BLENDER. *BLENDER - Working with Autodesk FBX files*. 2016. Disponível em: <https://wiki.blender.org/index.php/Extensions:2.6/Py/Scripts/Import-Export/Autodesk{_}>. Acesso em: 14 de set. 2016. Citado na página 35.

CODESYS. *CODESYS OPC Server*. Disponível em: <https://opcfoundation.org/products/view/codesys-opc-server>. Acesso em: 22 de set. 2016. Citado na página 47.

DANEELS, A.; SALTER, W. WHAT IS SCADA ? p. 339–343, 1999. Citado na página 19.

DERAKHSHANI, D.; MUNN, R.; MCFARLAND, J. *Introducing 3ds Max® 9 3D FOR BEGINNERS*. [S.l.]: Wiley Publishing, 2007. 555 p. ISBN 9780470097618. Citado na página 35.

DZIDEK, W. J.; ARISHOLM, E.; BRIAND, L. C. A realistic empirical evaluation of the costs and benefits of UML in software maintenance. *IEEE Transactions on Software Engineering*, v. 34, n. 3, p. 407–432, 2008. ISSN 00985589. Citado na página 39.

FILHO, M. E. M. et al. An integrated development model for character-based games. *SBGAMES2009 - 8th Brazilian Symposium on Games and Digital Entertainment*, p. 191–196, 2009. Citado na página 24.

FORBES. *Ford Pilots A New Virtual Assembly Plant To Build Its Cars*. 2014. Disponível em: <http://www.forbes.com/sites/unify/2014/03/31/ford-pilots-a-new-virtual-assembly-plant-to-build-its-cars/>. Acesso em: 17 de set. 2016. Citado 3 vezes nas páginas 6, 11 e 12.

HAMID, N. S. S.; AZIZ, F. A.; AZIZI, A. Virtual reality applications in manufacturing system. *Proceedings of 2014 Science and Information Conference, SAI 2014*, p. 1034–1037, 2014. Citado na página 15.

JAMRO, M.; TRYBUS, B. IEC 61131-3 programmable human machine interfaces for control devices. *2013 6th International Conference on Human System Interactions, HSI 2013*, n. Vm, p. 48–55, 2013. Disponível em: <http://www.scopus.com/inward/record.url?eid=2-s2.0-84883693786{&}partnerID=40{&}md5=ca90bf14cd9edb724851ad5599>. Citado 3 vezes nas páginas 6, 26 e 27.

JUNIOR, E. S. Tutorial sobre XNA – Parte 1 – Primeiros passos. v. 416748, p. 1–8, 2011. Citado 3 vezes nas páginas 6, 25 e 26.

KEBO, V.; STAŠA, P. Mining processes control and virtual reality. *Proceedings of the 2012 13th International Carpathian Control Conference, ICCC 2012*, p. 274–277, 2012. ISSN 1314-2704. Citado 3 vezes nas páginas 6, 21 e 22.

KING, A.; HUNT, R. Protocols and architecture for managing TCP/IP network infrastructures. *Computer Communications*, v. 23, n. 16, p. 1558–1572, 2000. ISSN 01403664. Citado na página 23.

Kontron Czech. *What is OPC?* 2011. Disponível em: <http://shop.kontron-czech.com/InfoPage.asp?TP=FT{&}ID>. Acesso em: 13 de set. 2016. Citado na página 23.

LASI, H. et al. Industry 4.0. *Business and Information Systems Engineering*, v. 6, n. 4, p. 239–242, 2014. ISSN 18670202. Citado na página 11.

LATTA, J. N.; OBERG, D. J. A Conceptual Virtual Reality Model. *IEEE Computer Graphics and Applications*, 1994. ISSN 02721716. Citado na página 15.

LOEFFLER, C. E.; ANDERSON, T. *The Virtual reality casebook*. New York: Van Nostrand Reinhold, 1994. xxv, 357 p. p. ISBN 0442017766. Citado 2 vezes nas páginas 6 e 16.

MAHMOUD, M. S.; SABIH, M.; ELSHAFEI, M. Using OPC technology to support the study of advanced process control. *ISA Transactions*, Elsevier, v. 55, p. 155–167, 2015. ISSN 00190578. Disponível em: <http://dx.doi.org/10.1016/j.isatra.2014.07.013>. Citado na página 24.

METLAB. *METLAB - Exporting FBX Files from 3DS Max*. 2016. Disponível em: <http://metlab.cse.msu.edu/exporting.html>. Acesso em: 14 de set. 2016. Citado na página 35.

MUJBER, T. S.; SZECSI, T.; HASHMI, M. S. J. Virtual reality applications in manufacturing process simulation. *Journal of Materials Processing Technology*, v. 155-156, n. 1-3, p. 1834–1838, 2004. ISSN 09240136. Citado 2 vezes nas páginas 6 e 15.

NETTO, A. V.; MACHADO, L. d. S.; OLIVEIRA, M. C. F. de. *Realidade Virtual - Fundamentos e Aplicações*. Florianópolis: [s.n.], 2002. 53 - 76 p. ISBN 85-7502-082-X. Citado 3 vezes nas páginas 6, 17 e 18.

OKSANEN, T.; PIIRAINEN, P.; SEILONEN, I. Remote access of ISO 11783 process data by using OPC Unified Architecture technology. *Computers and Electronics in Agriculture*, Elsevier B.V., v. 117, n. 2015, p. 141–148, 2015. ISSN 01681699. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S0168169915002252>. Citado na página 24.

OTTO, A.; HELLMANN, K. IEC 61131: A general overview and emerging trends. *IEEE Industrial Electronics Magazine*, v. 3, n. 4, p. 27–31, 2009. ISSN 19324529. Citado 2 vezes nas páginas 27 e 28.

PORTO, A. J. V. et al. MANUFATURA VIRTUAL: CONCEITUAÇÃO E DESAFIOS. 2002. Citado na página 17.

RODRÍGUEZ, F. et al. proposal for teaching SCADA systems proposal for teaching SCADA systems proposal for teaching SCADA systems proposal for teaching SCADA systems using Virtual in using Virtual Virtual in using Engineering Virtual Industrial Plants in in Education. p. 138–143, 2016. Citado na página 19.

ROHRER, M. W. Proceedings of the 2000 Winter Simulation Conference. p. 1211–1216, 2000. Citado na página 11.

RUMBAUGH, J.; JACOBSON, I.; BOOCH, G. *The UML reference manual*. [S.l.: s.n.], 1999. v. 1. 999 p. ISBN 020130998X. Citado na página 39.

RÜSSMANN, M. et al. Industry 4.0. The Future of Productivity and Growth in Manufacturing. *Boston Consulting*, n. April, 2015. Citado na página 11.

STEUER, J. Defining Virtual Reality: Dimensions Determining Telepresence. *Journal of Communication*, v. 42, n. 4, p. 73–93, 1992. ISSN 00219916. Disponível em: <http://doi.wiley.com/10.1111/j.1460-2466.1992.tb00812.x>. Citado na página 16.

TAN, V. V.; YOO, D.-S.; YI, M.-J. *An Approach to Remote Monitoring and Control based on OPC Connectivity*. IFAC, 2009. v. 42. 259–262 p. ISSN 14746670. Disponível em: <http://linkinghub.elsevier.com/retrieve/pii/S1474667016341696>. Citado na página 24.

THOMAS, M.; KUMAR, P.; CHANDNA, V. Design, development and commissioning of a human machine interface (HMI) laboratory for research & training. *IEEE Power Engineering Society General Meeting, 2004.*, v. 19, n. 3, p. 1582–1588, 2013. ISSN 0885-8950. Citado 2 vezes nas páginas 20 e 21.

TISSERANT, E.; BESSARD, L.; SOUSA, M. D. An open source IEC 61131-3 integrated development environment. *IEEE International Conference on Industrial Informatics (INDIN)*, v. 1, p. 183–187, 2007. ISSN 19354576. Citado na página 26.

YOUNAS, U.; DURRANI, S.; MEHMOOD, Y. Designing Human Machine Interface for Vehicle's EFI Engine Using Siemen's PLC and SCADA System. *Proceedings - 2015 13th International Conference on Frontiers of Information Technology, FIT 2015*, p. 205–210, 2016. Citado na página 29.

ZHENG, L. Z. L.; NAKAGAWA, H. OPC (OLE for process control) specification and its developments. *Proceedings of the 41st SICE Annual Conference. SICE 2002.*, v. 2, p. 917–920, 2002. Citado na página 22.

ZYDA, M. From visual simulation to virtual reality to games. *Computer*, v. 38, n. 9, p. 25–32, 2005. ISSN 0018-9162. Citado na página 11.