GUILHERME YUDI MAIBASHI
MARCOS MAKOTO SAITO

SISTEMA DE TELEOPERAÇÃO DE MANUFATURA VIA INTERNET

São Paulo
2009

GUILHERME YUDI MAIBASHI
MARCOS MAKOTO SAITO

SISTEMA DE TELEOPERAÇÃO DE MANUFATURA VIA INTERNET

Monografia apresentada a Escola Politécnica da Universidade de São Paulo referente à disciplina PMR2550 – Projeto de Conclusão de Curso II

Curso de Graduação
Engenharia Mecatrônica

Orientador:
Prof. Dr. Fabrício Junqueira

São Paulo
2009

Aos meus pais Sueli e Reginaldo
Guilherme Yudi Maibashi

Aos meus pais Lidia e Nelson
Marcos Makoto Saito

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DA ENGENHARIA MECÂNICA/NAVAL DA ESCOLA POLITÉCNICA (EPMN) – USP.

Maibashi, Guilherme Yudi
Sistema de teleoperação de manufatura via internet /
G.Y. Maibashi, M.M. Saito. -- São Paulo, 2009.
42 p. e 4 apêndices

Trabalho de Formatura - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos.

1. Manufatura 2. Internet I. Saito, Marcos Makoto II. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos III. t.

Agradecimentos

Ao nosso orientador, Prof. Dr. Fabrício Junqueira, pela sua constante orientação e incentivo para o desenvolvimento deste trabalho.

Ao aluno de doutorado José Garcia pelo apoio e incentivo.

À Escola Politécnica da USP, em especial ao Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos, que institucionalmente viabilizaram este trabalho.

SUMÁRIO

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

CIM – *Computer Integrated Manufacturing*

CLP – Controlador Lógico Programável

CP – Controlador Programável

SED – Sistema a Eventos Discretos

SP – Sistema Produtivo

Resumo

Tendo em vista o crescente interesse na utilização da comunicação via internet nos mais variados setores da economia, devido ao aumento da segurança e seu baixo custo em relação a outros meios de transporte de dados, faz-se possível o desenvolvimento de aplicativos web que apresentem soluções para o monitoramento e controle de processos e sistemas remotos. Neste trabalho, tem-se como objetivo projetar e implementar um sistema que possibilite controlar e monitorar remotamente um equipamento industrial via internet. Para desenvolver este trabalho, será utilizada a estação de inspeção da célula de manufatura didática - Mini CIM (*Computer Integrated Manufacturing* - CIM) do laboratório de Sistemas de Automação. Para realização desta tarefa, conceitos de sistemas a eventos discretos (SED) e metodologia de projeto de controle são apresentados, assim como as tecnologias aplicadas, como Controladores Programáveis e PROFIBUS. Para a realização da teleoperação da estação de inspeção do Mini CIM, foi desenvolvido uma aplicação web que se comunica via *WebService* com o *software* de controle do Mini CIM. O *WebService* foi escolhido como meio de comunicação pois cada vez tem sido utilizado entre as empresas, por facilitar a integração entre diferentes sistemas, trazendo agilidade e confiança na troca de informação. Com a conclusão do projeto, a estação de inspeção do Mini CIM pode ser monitorada e controlada por um operador local ou remoto.

Palavras-Chave: manufatura, Internet, teleoperação.

Abstract

The increasing interest on the internet communication in several economy sectors, because the evolution on digital security and its low cost if compared to others ways to transfer data, it enables the development of web applications that presents solutions to monitoring and controlling of manufacturing process and remote systems. The aims of this work are design and implement a system that uses internet to provide remote control and monitoring of an industrial. To development this project, it was used the inspection station of a didactic manufacture cell (Mini CIM (Computer Integrated Manufacturing)) from the Automation Systems Laboratory as a study case. Concepts of discrete event systems (DES) and design methodology are presenting, even as the applied technology as Programmable Logic Controller (PLC) and PROFIBUS. To perform the teleoperation of the Mini CIM inspection station, it was developed a web application that communicates, with the inspection station software control by WebService. The WebService was chosen due its increasing application on the companies, providing agility and security on transfer information. At the project conclusion, the Mini CIM inspection station can be controlled and monitored by a local or remote operator.

Keywords: manufacturing, Internet, teleoperation.

1) Introdução

Sistemas produtivos (SPs) são sistemas que realizam processos para produção de bem ou serviços (VILLANI, 2003). Os grandes avanços na área de telecomunicação e transmissão de dados estão modificando a maneira de se conceber e produzir bens e serviços. Os meios de produção deixam de ser organizados em plantas únicas e lineares, para se tornarem organizados de forma fisicamente dispersa e com inúmeros processos que são conduzidos de modo paralelo. Isso leva à meios de produção mais complexos, com maior grau de automatização, com tecnologias programáveis e flexíveis, mecanismos autônomos e sistemas distribuídos fisicamente (JUNQUEIRA, 2006). A este novo paradigma de produção é conhecido como manufatura dispersa ou manufatura distribuída.

Por manufatura dispersa (ou distribuída) entende-se como uma combinação de conceitos emergentes voltados para o gerenciamento e controle distribuído da manufatura, que surgiu como tentativa de tratar e explorar a autonomia de componentes do sistema, visando torná-lo mais competitivo em um ambiente com facilidades de comunicação e diversidade de recursos distribuídos geograficamente (GOULART, 2000).

Além disso, a Internet possui grande facilidade de criação de ambientes gráficos, o que facilita a interface com o usuário com um custo relativamente baixo (ÁLVARES e SOUSA, 2001). Sendo uma rede de comunicação, é possível enviar e receber informações, que podem ser comandos para serem executados em algum dispositivo ligado à rede. Esse dispositivo pode ser robótico (TAYLOR e TREVELYAN, 1995) ou máquina-ferramenta CNC (KAO e LIN, 1996).

Desta maneira, há uma situação extremamente favorável para o desenvolvimento de um sistema de manufatura teleoperado via internet. Este trabalho tem assim o objetivo de desenvolver e implementar um sistema de controle e monitoração remota de um SP. O SP estudado neste trabalho é a estação de inspeção do Mini CIM localizado no Laboratório de Automação da Escola Politécnica da USP.

1.1) Objetivo do Trabalho

O objetivo deste trabalho é o projeto e implementação de um sistema de teleoperação e monitoração via internet. Como estudo de caso, utilizou-se a "estação de inspeção" da célula de manufatura didática (Mini CIM), o qual está instalado no laboratório de Sistemas de Automação da Escola Politécnica da USP.

1.2) Organização do Texto

No capítulo 2 apresenta-se a fundamentação teórica necessária para a execução deste trabalho, no qual são abordados os temas como sistemas produtivos (SP), controle e modelagem de sistemas a eventos discretos (SED), operação remota e tecnologias web.

No capítulo 3, aborda-se o sistema Mini CIM. Neste capítulo, descrevem-se as partes que compõem o sistema como, por exemplo, o CLP utilizado, o protocolo de comunicação PROFIBUS e as estações de trabalho como, por exemplo, a estação de inspeção.

No capítulo 4, aborda-se a etapa de projeto. Inicialmente, há uma apresentação da metodologia de projeto aplicada neste trabalho. Na sequência, apresenta-se a documentação gerada durante a elaboração do projeto. A execução do projeto é descrita na sequência do texto.

No capítulo 5 são apresentadas a conclusões e comentários finais do trabalho.

2) Fundamentação Teórica

2.1) Sistema Produtivo

Segundo GROOVER (2000) um Sistema Produtivo (SP) pode ser classificado como um conjunto de pessoas, equipamentos e procedimentos comprometidos para realizar as operações de manufatura de um processo.

O objetivo de um SP consiste na execução de tarefas, com a finalidade de agregar valor a um determinado produto. Segundo SANTOS FILHO (2000), a execução das tarefas pode ser interpretada de duas formas distintas:

- Transformação de estado do elemento que passa pelo SP;
- Prestação de serviço ao elemento que passa pelo SP.

O SP abordado neste trabalho é classificado como um sistema a eventos discreto (SED). Segundo KANEKO (2008), um sistema de controle a eventos discretos (SED) envolve um conjunto de dispositivos que, baseado num procedimento pré-estabelecido ou numa lógica fixa que estabelece um procedimento, executa ordenadamente cada estágio do controle.

Um evento pode ser a chegada de um consumidor em uma fila, a conclusão de uma tarefa ou falha de uma máquina em um sistema de manufatura, a transmissão de uma mensagem em uma rede de dados, a conclusão dos processos de um programa de computador, ou a variação de um valor de referência em um sistema de controle, etc. Logo, exemplos de SEDs também incluem muitos sistemas feitos pelo homem tais como os computadores, as redes de comunicação, a robótica e os sistemas de manufatura, programas de computador, etc (MIYAGI, 1996).

Um SED possui um conjunto de estados discretos e que, ao contrário de um sistema físico, pode utilizar valores simbólicos ao invés de valores reais. Por exemplo, uma máquina pode ser representada como ociosa, trabalhando ou quebrada. Diferentemente da maioria dos sistemas físicos, a relação entre a transição de estado e os eventos são altamente irregulares e, usualmente, não podem ser descritas utilizando equações diferenciais ou de diferenças (MIYAGI, 1996).

2.2) Controle de Sistemas Produtivos

Segundo CASSANDRAS (1992), sistemas de controle têm o objetivo de manter um comportamento desejado, em relação aos sinais de entrada, em um determinado sistema.

Os sistemas de controle podem ser classificados em três diferentes categorias:

- Sistema de Controle de Variáveis Contínuas (SVC): Este sistema de controle trabalha com variáveis quantitativas que podem assumir infinitos valores dentro de certos intervalos determinados e, normalmente, é usado para controle de grandezas físicas analógicas e/ou contínuas como vazão, pressão, temperatura, etc (CAVALHEIRO, 2004);
- Sistema de Controle de Variáveis Discretas (SED): Este sistema de controle trabalha com estados discretos e eventos instantâneos, isto é, trabalha com variáveis qualitativas que assumem valores determinados, usualmente 0 ou 1. Mas, esses valores não variam continuamente no tempo (CAVALHEIRO, 2004);
- Sistema Híbridos: Este sistema de controle trata, em conjunto, as duas classes de controle apresentadas anteriormente, SVC e SED (MIYAGI, 1996).

No controle de SED, as operações fundamentais são realizadas por dispositivos tais como operadores lógicos, operadores aritméticos e temporizadores, que possuem sinais de entrada e/ou saída que podem assumir valores discretos (MIYAGI, 1996).

2.3) Técnicas de Modelagem

A rede de Petri é uma técnica efetiva para a modelagem de sistemas. A sua base teórica permite o desenvolvimento de poderosas técnicas e ferramentas de análise e síntese de estratégias de controle. Considera-se que na modelagem inicial sejam utilizadas interpretações (inscrições) em linguagem

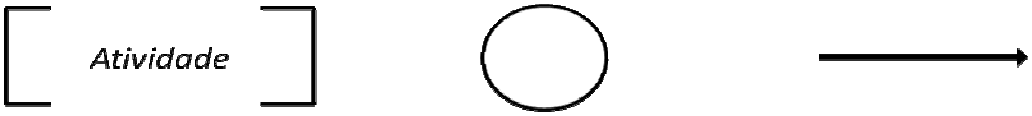
natural (não formais) e a partir deste modelo desenvolver um detalhamento gradativo com interpretações mais específicas (formais) (MIYAGI, 1996).

Neste capítulo apresenta-se a técnica do PFS (*Production Flow Schema*) e do MFG (*Mark Flow Graph*) que são versões da rede de Petri próprias para aplicação em diferentes níveis de modelagem de SED.

2.3.1) *Production Flow Schema* (PFS)

Na modelagem inicial de um sistema real, considera-se a divisão do sistema em um pequeno número de partes discretas, pois a identificação destas partes deve facilitar a compreensão do sistema. Um SED pode ser caracterizado com base no fluxo de itens (“coisas”) e desta forma, qualquer processo produtivo pode ser decomposto em três elementos básicos (MIYAGI, 1996):

- *Elemento Ativo:* correspondentes a atividades (Figura 1a);
- *Elemento Distribuidor:* correspondentes a distribuições (Figura 1b);
- *Arco*: representam as relações entre os componentes anteriores (Figura 1c).



(a) (b) (c)

Figura 1 – Elementos PFS: (a) Elemento Ativo (Atividade); (b) Elemento Distribuidor; e (c) Arco

2.3.2) *Mark Flow Graph* (MFG)

O *Mark Flow Graph* (MFG), que é apresentado a seguir, é um grafo derivado da rede de Petri onde as funções de entrada e saída, a propriedade de "safeness", livre de contato, etc, são devidamente consideradas, visando a modelagem e o controle dos SEDs de forma mais simples e eficaz. O MFG é composto pelos seguintes elementos estruturais (MIYAGI, 1996):

- **Box**: indica uma condição e é representado por um bloco quadrado (Figura 2a);

- **Transição**: indica um evento e é representado por uma barra vertical (Figura 2b);
- **Arco Orientado**: conecta **boxes** e **transições** para indicar a relação entre uma condição e os pré e os pós-eventos que o definem. É representado por uma seta. Os **arcos de saída** são os **arcos** que saem de um **box** ou **transição**, isto é, estão conectados no lado de saída destes elementos; **arcos de entrada** são **arcos** que entram em um **box** ou **transição**, isto é, estão conectados no lado de entrada destes elementos (Figura 2c);
- **Marca**: indica a manutenção de uma condição e é representada por um ponto negro no interior do **box** correspondente a esta condição (Figura 2d);
- **Porta**: habilita ou inibe a ocorrência dos eventos correspondentes às **transições** sendo denominada **porta habilitadora** ou **porta inibidora**, conforme sua natureza. Estas, por sua vez, podem ser subclassificadas em **porta externa** ou **porta interna**, de acordo com a origem do sinal. A **porta habilitadora** (Figura 2e) é uma porta que possui um círculo negro na extremidade conectada à **transição**. Quando o sinal de origem for "1", esta porta **habilita** a **transição** em que está conectada, compondo um AND lógico com as outras condições que determinam a ocorrência do evento correspondente. A **porta inibidora** (Figura 2f) é uma porta que possui um círculo branco na extremidade conectada à **transição**. Quando o sinal de origem for "1", esta porta inibe a **transição** em que está conectada, compondo um OR lógico com as outras condições que determinam a ocorrência do evento correspondente. A origem do sinal de uma **porta interna** é um **box**. Quando existir **marcas** no **box**, o sinal é "1"; quando não existir **marcas** é "0". A origem do sinal de uma **porta externa** não faz parte do grafo, ou seja, ela indica a entrada de um sinal binário gerado por algum dispositivo externo;
- **Arco de sinal de saída**: este **arco** envia um sinal binário do **box** para os dispositivos externos do grafo e é representado por uma

linha que conecta estes dois elementos. Quando houver uma **marca** neste **box**, o sinal é "1"; quando não houver, é "0". (Figura 2g).

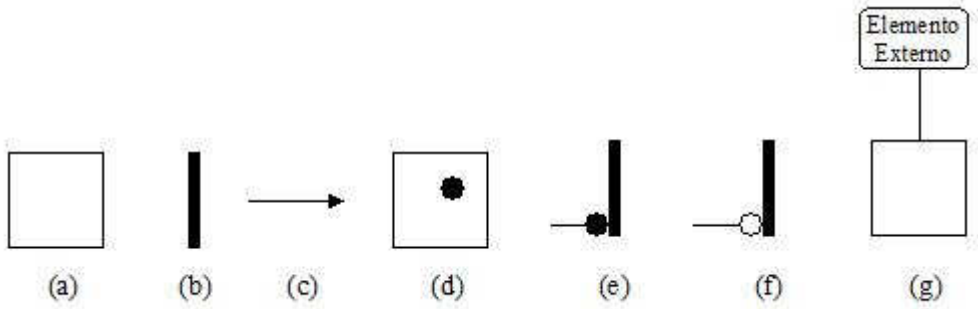


Figura 2 – Elementos Básicos do MFG

2.4) Linguagem de Controle de SED

A linguagem de controle de SED é a forma de descrever concretamente os comandos, para que o dispositivo de controle execute controle do sistema conforme a especificações. A ferramenta mais popular nesta área é o diagrama elétrico de relés, que tem sido utilizado como um sinônimo de linguagem de controle de sistemas sequenciais (MIYAGI, 1996).

2.4.1) Diagrama de Relés (*Ladder Diagram*)

O diagrama de relés (*Ladder*) está presente praticamente em todos os CLPs disponíveis no mercado. Por ser uma linguagem gráfica, baseada em símbolos semelhantes aos encontrados nos esquemas elétricos (contatos e bobinas), as possíveis diferenças existentes entre os fabricantes de CLPs, quanto à representação das instruções, são facilmente assimiladas pelos usuário (CASTRO, 2007).

O diagrama de relés possui regras para posicionar e conectar elementos como contactos e bobinas, e também regulamenta o fluxo e o processamento de sinais. As regras do diagrama de relé (MIYGI, 1996) são:

- Os contactos devem ficar na intersecção das linhas e colunas de uma matriz e as bobinas devem ocupar somente a última coluna à direita;

- As linhas verticais das extremidades à direita e à esquerda chamam-se linhas mãe; na da esquerda são conectadas somente as bobinas;
- Os contactos e as bobinas são conectados por meio de linhas horizontais; as linhas horizontais são interligadas por meio de linhas verticais e não se permitem várias linhas em uma única coluna; a intersecção entre uma linha horizontal e uma linha vertical pode ser uma conexão ou apenas um cruzamento sem conexão.

As regras para o fluxo e processamento de sinais são:

- A energia flui através das linhas horizontais da esquerda para a direita e de acordo com os estados aberto/fechado dos contactos; executa-se a função lógica AND;
- A energia flui através das linhas verticais de cima para baixo; a linha vertical executa a função lógica OR dos estados das linhas horizontais que estão à esquerda, transmitindo o resultado para a(s) linha(s) horizontal(is) à sua direita;
- O acionamento das bobinas depende da existência de fluxo de energia da linha horizontal à sua esquerda;
- O processamento do diagrama de relés é realizado de cima para baixo.

2.5) Operação Remota

A teleoperação é definida (TOURINO, 2000) como o controle contínuo e direto de um teleoperador (máquina remota). Ela pode ser classificada (ZHAI, 1991), como:

- Controle manual sem auxílio computacional;
- Controle manual com significativo auxílio ou transformação computacional;
- Controle supervisório com predomínio do controle realizado pelo operador humano;

- Controle supervisório com predomínio do controle realizado pelo computador;
- Controle completamente automático, onde os operadores humanos observam o processo sem interferir.

A teleoperação proporciona para as indústrias agilidade e flexibilidade, uma vez que o operador não necessita estar presente fisicamente no local para poder executar alterações nas funções (KANEKO, 2008).

Prevenir, deter, detectar e responder são as ações que definem a importância de uma monitoração remota em um sistema que envolve uma rede de comunicação (ROCHA, 2003).

Segundo SOUZA (2002), as redes de comunicação estão presentes na sociedade, interligando desde computadores pessoais em uma casa até os dispositivos computacionais de uma empresa. Por facilitar o desenvolvimento de trabalhos e permitir o acesso a informações *on line* na internet, as redes de comunicação são cada vez mais objeto de novos serviços oferecidos pelas empresas e instituições, por isso torna-se vital mantê-las sempre disponíveis e com bom desempenho.

A monitoração remota é empregada em diversos casos, desde o apoio para as operações de máquinas ferramentas até em serviços de bancos e hospitais (CHRIST *et al.*, 2006).

2.6) Tecnologias WEB

A internet surgiu de pesquisas militares na época da guerra fria. Já na década de 70, com o fim da tensão que existia entre os EUA e a União Soviética, as universidades começaram a ter acesso aos estudos na área. E a partir de 1990 a internet começou a ser mais divulgada e com isso sua popularidade não parou de crescer. Em 2008, o número de usuários pelo mundo já passam dos 1,5 bilhões de acordo com a INTERNET USAGE WORLD STATISTICS (INTERNET USAGE WORLD STATISTICS, 2009).

A internet tem se tornado uma solução muito interessante na área de teleoperação, principalmente devido ao seu baixo custo em comparação ao uso

de equipamentos teleoperados, ao fácil desenvolvimento de interfaces gráficas na rede e a possibilidade de se conectar em qualquer lugar com acesso a internet (ÁLVARES *et al*., 1999 e 2000). Com o desenvolvimento dos mais variados serviços de internet, estes exigem a criação de métodos que possibilitem o processamento de dados tanto do lado servidor como do lado cliente. Neste caso, pode-se citar algumas tecnologias como o CORBA, o DCOM, a RMI. Cada uma destas tecnologias possui suas limitações e para resolver o problema surgiu o *WebService* (CAMELO, 2002).

2.7) WebServices

O *WebService* é qualquer tipo de serviço disponibilizado na internet que use o empacotamento XML (*eXtensible Markup Language*) e não seja amarrado a nenhum sistema operacional ou linguagem de programação. Além disso, não é necessário, mas é desejável que ele tenha mais duas propriedades (CERAMI, 2002):

- Auto-explicativo: para facilitar a integração de outros sistemas com o *WebService* é interessante que ele tenha sua documentação disponibilizada, contendo, por exemplo, uma lista com seus métodos, parâmetros e valores retornados;
- Localizável: quando um *WebService* é criado, é desejável que se tenha uma forma de publicá-lo, e para isso podem ser utilizados diversos sistemas de buscas.

A estrutura de um *WebService* pode ser separada em quatro camadas (CERAMI, 2002): serviço de transporte, mensagem XML, descrição de serviço e descoberta de serviço.

A camada de serviço de transporte é responsável pelo transporte da mensagem entre as aplicações. Atualmente, esta camada inclui HTTP (*HyperText Transfer Protocol*), SMTP (*Simple Mail Transfer Protocol*), FTP (*File Transfer Protocol*) e protocolos mais novos como BEEP (*Blocks Extensible Exchange Protocol*) (CERAMI, 2002).

Na camada da mensagem, o XML foi escolhido para a realização de troca de informação dos *WebServices* pois possibilita que diferentes sistemas possam trocar informações, independente do sistema operacional ou da linguagem de programação.

Para a comunicação têm-se dois protocolos padrões (CERAMI, 2002):

- *Simple Object Access Protocol* (SOAP): Este protocolo também se utiliza do XML para a troca de informação, seu foco é o transporte de RPCs (*Remote Procedure Calls*) via HTTP, apesar de poder ser utilizado nos mais variados sistemas de mensagens;
- XML-RPC: Protocolo que utiliza mensagens XML para realizar RPCs. Os pedidos são codificados em XML e enviados via HTTP POST (método de envio de informação HTTP). A resposta vem contida no corpo da resposta HTTP. Ele é mais simples que o SOAP e mais fácil de ser adotado, entretanto ele não suporta chamadas automáticas, o que impede sua integração em aplicações "just-in-time".

A camada da descrição do serviço é responsável por descrever a interface pública de um *WebService* específico, ela é descrita pelo WSDL (*Web Service Description Language*).

O WSDL utiliza XML para especificar uma interface pública. Esta interface pública pode conter informações de todas as funções e dados publicáveis, vinculando informações sobre o protocolo de transporte utilizado e o endereço específico para localizar o serviço (CERAMI, 2002).

A camada de descoberta de serviço é responsável por centralizar serviços em um registro comum e facilitar a publicação/busca por funcionalidades, atualmente isso é feito pelo UDDI (*Universal Description, Discovery, and Integration*).

3) Célula de manufatura didática

Computer Integrated Manufacturing (CIM) é um conceito de manufatura no qual o processo de produção é controlado por computador. A implantação deste método de manufatura exige o controle do processo em malha fechada com o monitoramento dos sensores da planta de processo em tempo real.

3.1) Mini CIM

Mini CIM é um pequeno sistema de manufatura integrado ao computador (*Computer Integrated Manufacturing* - CIM), uma célula de manufatura didática, que visa simular o processo de montagem de diferentes peças. O Mini CIM presente no Laboratório de Sistemas de Automação consiste de 4 estações de treinamento, cada qual capaz de ser operada individualmente ou integrada com as demais estações.

- Estação de Alimentação ou Distribuição: trata-se do sistema responsável por armazenar as peças, separa uma peça do magazine de distribuição e disponibilizar a peça para o processo inspeção;
- Estação de Inspeção ou Testes: trata-se do sistema responsável pela execução de testes com a aquisição de informação e comparação de características específicas;
- Estação de Montagem com Unidade de Execução da Montagem: trata-se do sistema responsável por montar um conjunto de peças em um produto determinado pelo usuário;
- Sistema Inteligente de Transporte (SIT): trata-se do sistema responsável por transportar as peças entre as estações de trabalho.

Nele podem ser realizados vários tipos de treinamento, desde pneumática e programação de CLP´s, até o estudo de protocolos de comunicação industrial, como o PROFIBUS.

Uma representação esquemática do Mini CIM pode ser visualizada na Figura 3.

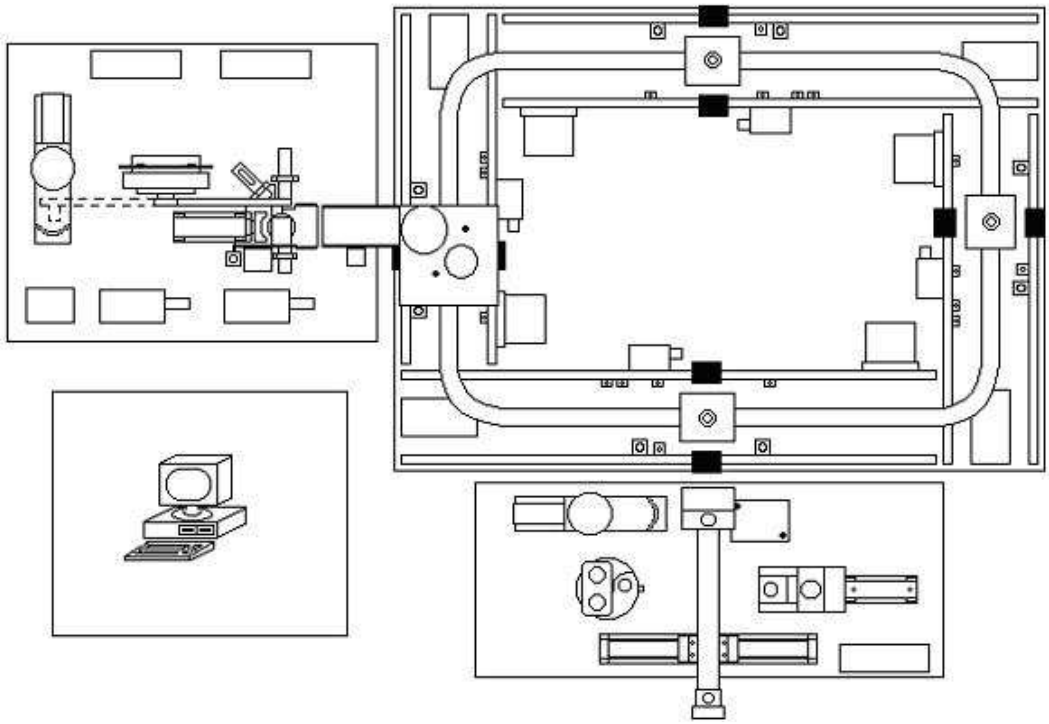


Figura 3 – Esquema Ilustrativo do Mini CIM

Neste trabalho, o objeto de estudo é a estação de inspeção do Mini CIM, e esta será detalhada no tópico a seguir. Serão abordados também o PROFIBUS, que é protocolo de comunicação utilizado entre os dispositivos do Mini CIM, e o CLP empregado no controle das I/O's da estação de inspeção.

3.2) Estação de Inspeção

Componentes importantes na execução de testes são a aquisição de informação e a comparação de características específicas e, resultante disto, uma decisão entre "peça aceita/rejeitada". Por exemplo, a checagem de disponibilidade, identidade, forma, dimensões, cor e peso ou checagem da orientação de uma peça.

As funções da estação de inspeção são:

- Estabelecer as características do material da peça;
- Checar as peças;
- Descartar ou disponibilizar uma peça para a estação subsequente.

Esta estação de testes apresenta uma plataforma elevatória e diferentes tipos de sensores para a realização dos testes. Para tanto, esta estação dispõe de dois “andares”. No “andar” inferior, existem três sensores distintos: indutivo, óptico e capacitivo, que estão dispostos de forma a não interferir na livre movimentação tanto da plataforma elevatória como do braço articulado do módulo de transferência (estação de distribuição). O sensor óptico é o único que permanece acoplado à plataforma elevatória, sendo os demais fixos na base da unidade. O sensor indutivo visa identificar peças metálicas (prateadas); o sensor óptico destina-se a identificar peças que refletem a luz por ele emitida (rosas ou prateadas); e o sensor capacitivo é utilizado para identificar a presença de peças. Os três sinais devem ser analisados em conjunto, pois nenhuma outra combinação de sinais consegue identificar o tipo/material da peça corretamente. A Figura 4 ilustra como estão montados os sensores na estação de inspeção.

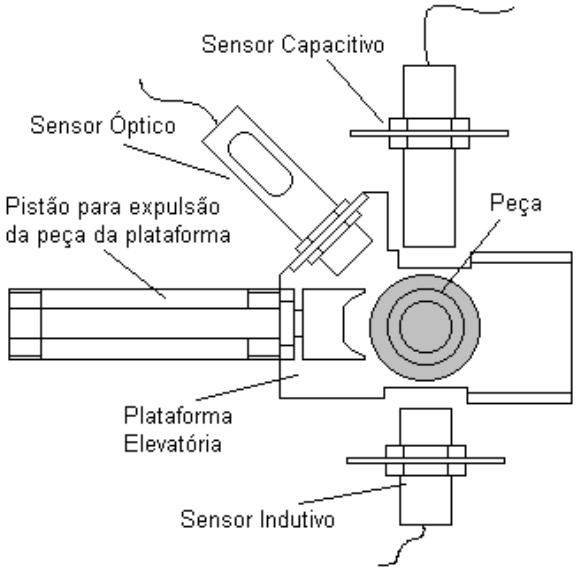


Figura 4 – Arranjo físico dos sensores do material das peças na Estação de Inspeção

O “andar” superior, onde será realizado o teste de altura das peças, apresenta um cilindro pneumático e um mecanismo acoplado à extremidade de sua haste, cujo movimento pressiona um sensor piezelétrico acoplado à estrutura, na posição vertical. O contato com a peça é realizado por uma pequena haste metálica, que é comprimida contra a peça a ser testada, em um movimento ao longo de seu eixo, gerando a compressão do cristal piezelétrico.

A compressão do cristal piezelétrico gera uma pequena corrente elétrica. A altura da peça pode ser testada de acordo com a intensidade da corrente gerada deste contato (maior para compressões maiores - peças mais altas - ou menores para menores compressões - peças mais baixas), isto é, da calibração deste sensor.

A plataforma elevatória conta com mais um cilindro pneumático, na posição horizontal, que também se desloca para os diferentes “andares” (acoplado à plataforma elevatória), responsável por expulsar as peças, tanto aprovadas como reprovadas no teste de altura, por meio de rampas presentes em cada um dos andares: a rampa do “andar” superior para peças aprovadas e a do “andar” inferior para peças reprovadas.

Após a realização dos testes, as peças aprovadas seguem para um carro transportador no sistema inteligente de transporte (SIT) ou são descartadas no andar inferior, conforme pode ser visto na Figura 5.

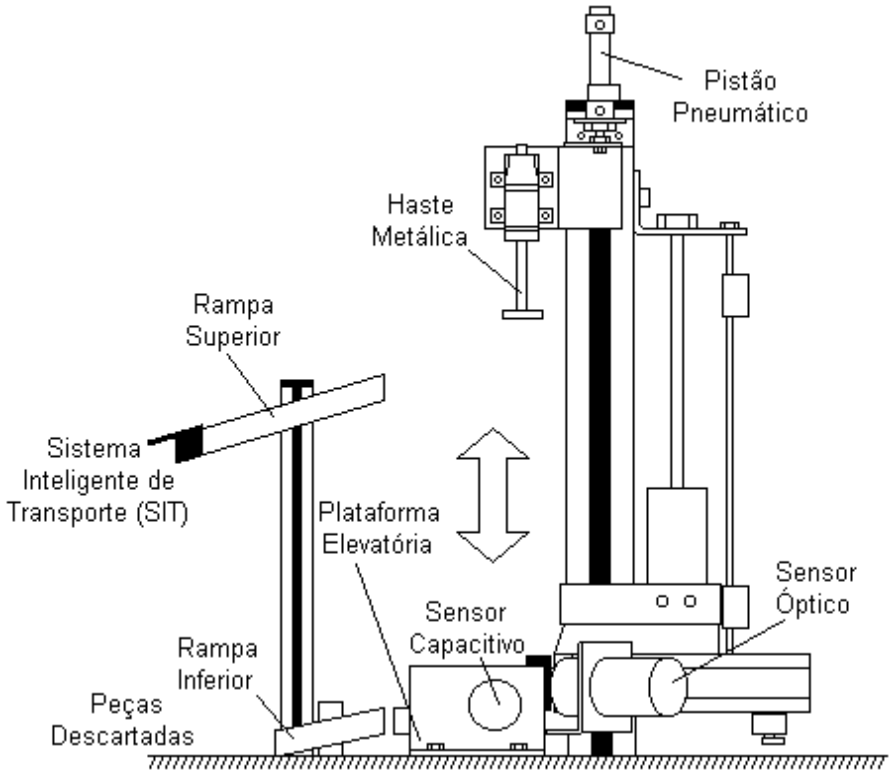


Figura 5 – Plataforma elevatória e rampas inferior e superior

3.3) PROFIBUS

O PROFIBUS é um padrão de rede de comunicação de campo, aberto e independente de fornecedores, no qual a interface entre dispositivos permite uma ampla aplicação em processos de manufatura e automação predial. Este protocolo pode utilizar como meio de transmissão qualquer um dos seguintes padrões: RS-485, ISC 61158-2 ou fibra óptica (KANEKO, 2008).

O PROFIBUS começou como um projeto apoiado por autoridades públicas, que iniciou em 1987 na Alemanha. Dentro do contexto deste projeto, 21 companhias e institutos uniram forças e criaram um projeto estratégico FIELDBUS. O objetivo era a realização e manutenção de um barramento de campo *bitserial*, sendo o requisito básico a padronização da interface do dispositivo de campo. Por esta razão, os membros das companhias do ZVEI (Associação Central da Indústria Alemã) concordaram em adotar um conceito técnico para manufatura e automação de processos (PROFIBUS, 2006).

3.3.1) PROFIBUS DP – Periferia Descentralizada

De acordo com KANEKO (2008), o PROFIBUS especifica as características técnicas e funcionais de um sistema de comunicação industrial. Por meio dos dispositivos digitais podem se interconectar, do nível de campo, com os CPs, até o nível de sensores, atuadores, válvulas, etc.

Dispositivos mestres determinam a comunicação de dados no barramento. Um mestre pode enviar mensagens, sem uma requisição externa, sempre que possuir o direito de acesso ao barramento. Os mestres também são chamados de estações ativas no protocolo PROFIBUS.

Os dispositivos escravos são dispositivos remotos (de periferia), tais como módulos de I/O, válvulas e transdutores. Eles não têm direito de acesso ao barramento e só podem enviar mensagens ao mestre ou reconhecer mensagens recebidas quando solicitados. Os escravos também são chamados estações passivas.

O PROFIBUS é um sistema multimestre, ou seja, permite a operação conjunta de diversos sistemas de automação, engenharia e/ou visualização,

com os seus respectivos dispositivos periféricos (por exemplo: I/O's). Ele diferencia seus dispositivos entre mestres e escravos.

O PROFIBUS-DP, otimizado para alta velocidade de transmissão e conexão de baixo custo, foi projetado especialmente para a comunicação entre sistemas de controle de automação e seus respectivos I/O's distribuídos no nível de dispositivo (KANEKO, 2008).

3.4) Meio de transmissão RS-485

O padrão RS485 é a tecnologia de transmissão mais frequentemente encontrada no PROFIBUS. Sua aplicação inclui todas as áreas nas quais uma alta taxa de transmissão aliada a uma instalação simples e barata são necessárias. Um par trançado de cobre blindado com um único par condutor é o suficiente neste caso (KANEKO, 2008).

A tecnologia de transmissão RS485 é muito fácil de utilizar. O uso de par trançado não requer nenhum conhecimento ou habilidade especial. A topologia por sua vez permite a adição e remoção de estações, bem como uma colocação em funcionamento do tipo passo-a-passo, sem afetar outras estações. Expansões futuras, portanto, podem ser implementadas sem afetar as estações já em operação (KANEKO, 2008).

Taxas de transmissão entre 9,6 kbit/s e 12 Mbit/s podem ser selecionadas, porém uma única taxa de transmissão é selecionada para todos os dispositivos no barramento, quando o sistema é inicializado (PROFIBUS, 2006).

3.5) Controladores Programáveis

No controle de SED, as operações fundamentais são realizadas por dispositivos como operadores lógicos, operadores aritméticos e temporizadores. Estes dispositivos possuem sinais de entrada e/ou saída que podem assumir valores discretos. Os Controladores Programáveis (CP) são providos de todas estas operações básicas de controle e representam o próprio dispositivo de realização do controle ou, pelo menos, a parte principal do sistema de controle. Como dispositivo de realização do controle é fundamental

para a execução das atividades do dispositivo de controle, a modelagem deste pode ser baseada na estrutura do CP.

A operação do CP está baseada em um processamento cíclico ilustrado na Figura 6. Em cada ciclo, primeiramente as entradas são coletadas, processadas e por fim, os resultados obtidos são enviados às saídas (MIYAGI, 1996).

Figura 6 – Ciclo de processamento do CP

Os controladores lógico programáveis (CLP) são equipamentos eletrônicos utilizados em sistemas de automação flexível. São ferramentas de trabalho muito úteis e versáteis para aplicações em sistemas de acionamentos de controle, e por isso, são utilizados em grande escala no mercado industrial. Permitem desenvolver e alterar facilmente a lógica para acionamentos das saídas em função das entradas. Desta forma, pode-se associar diversos sinais de entrada para controlar diversos atuadores ligados nos pontos de saída (SILVA FILHO, 1999). A Figura 7 esquematiza as inúmeras aplicações de um CLP.

3.5.1) CLP – WinAC 412

O CLP utilizado neste projeto é o Siemens WinAC 412, que pertence à família dos CPU 41x-2 PCI. Este CLP tem o formato similar a uma placa de computador, sendo compatível com o *slot* PCI de computador comum. Este CLP permite monitorar e controlar múltiplas I/O's em ambiente de Ethernet Industrial sob o protocolo PROFIBUS DP (SIEMENS, 2004) (Figura 8).

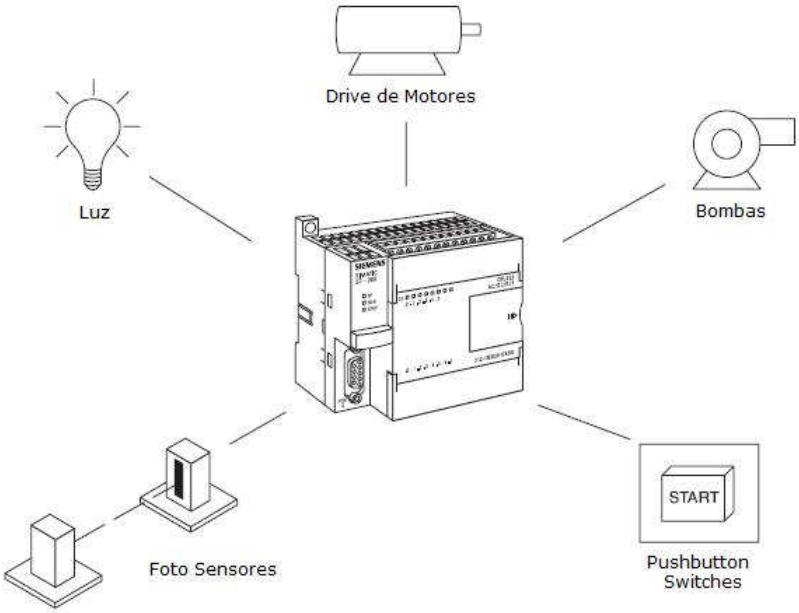


Figura 7 – Aplicações de um CLP (SIEMENS, 2004)

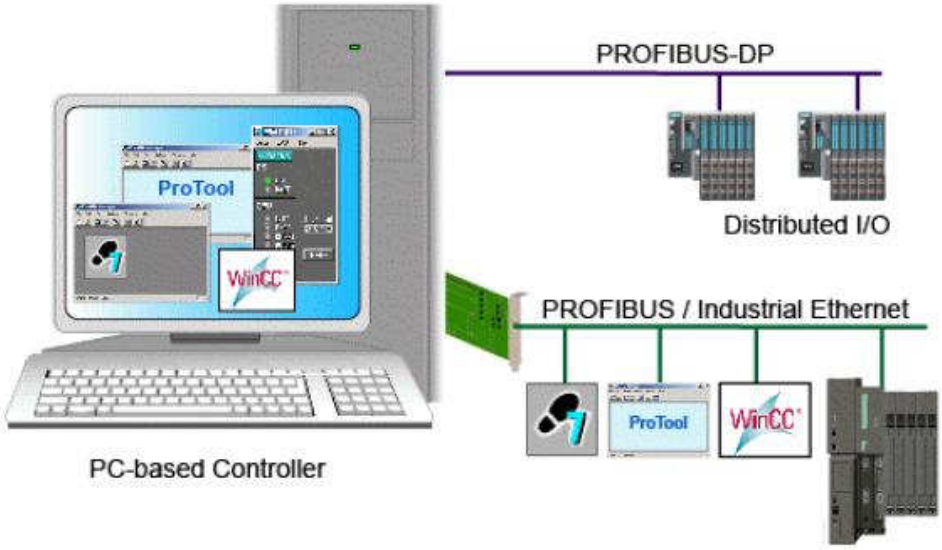


Figura 8 – Esquema da instalação do CLP WinAC 412

4) Projeto

Neste capítulo, inicialmente é apresentada a metodologia de projeto utilizada, explicando-se cada etapa para melhor desenvolvimento deste trabalho. Em seguida é abordada a execução da metodologia de projeto em si, com a apresentação de tabelas e diagramas PFS/MFG referentes ao SP estudado.

4.1) Metodologia

O desenvolvimento deste aplicativo é baseado na metodologia proposta por MIYAGI (1996), no qual o conceito de modularização é fundamental para a sistematização do desenvolvimento do sistema de controle. A Figura 9 esquematiza essa metodologia.

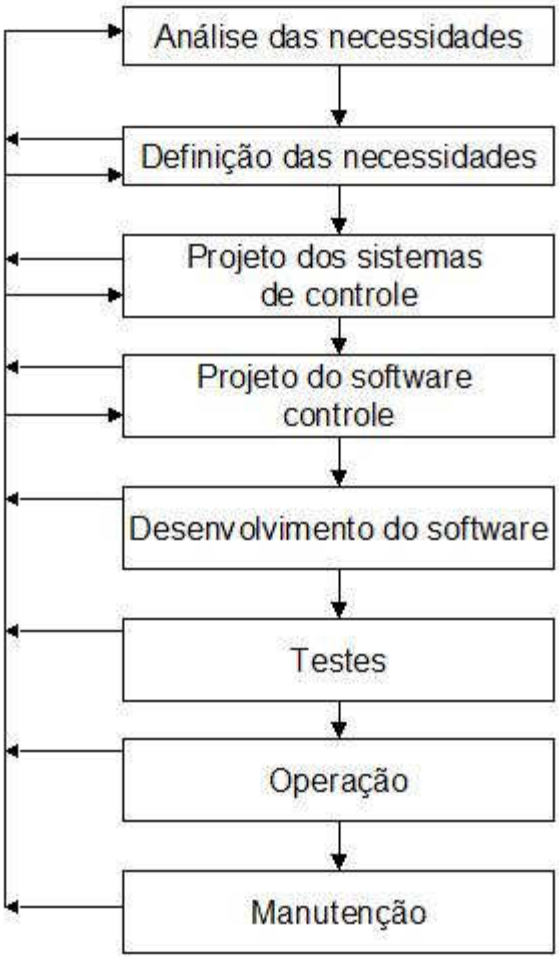


Figura 9 – Etapas da metodologia adotada

A metodologia deve considerar as diferentes abordagens em relação ao tipo do objeto de controle e ao porte do sistema. Dentro deste contexto também devem ser consideradas as técnicas de reutilização com aplicação de IA (inteligência artificial), técnicas de simulação de operações na etapa de projeto, técnica do MFG/PFS, etc., que possibilitam o projeto estruturado, técnicas com entrada gráfica (esquemática) do procedimento das operações da máquina, etc.

O conteúdo dos procedimentos nos sistemas de controle envolve as seguintes atividades:

1. Identificação do objetivo final do sistema;
2. Compreensão do objetivo de controle, instalações e equipamentos;
3. Organização dos conhecimentos sobre o sistema de controle (dispositivo de controle, equipamentos periféricos, etc.);
4. Abstração e análise das funções de controle, como os modos de operação e monitoração das instalações e equipamentos;
5. Definição das funções de controle;
6. Definição do fluxo das funções de controle;
7. Divisão das funções e definição das interfaces;
8. Definição e alocação dos sinais de entrada e saída;
9. Definição da estrutura do programa de controle;
10. Projeto de reutilização;
11. Projeto do(s) programa(s);
12. Projeto de programas não padronizados;
13. Desenvolvimento do programa e seu carregamento nas máquinas;
14. Teste por unidade;
15. Teste do sistema.

Relacionando estes procedimentos com o ciclo de vida do sistema de controle, (1) a (4) compõem a etapa de análise de necessidades, (5) e (6) a etapa de definição das necessidades, (7) a (9) a etapa de projeto do *software*

de controle, (13) a etapa de desenvolvimento do *software*, (14) e (15) a etapa de testes.

Desta forma, a metodologia apresentada é um meio de atender as necessidades do usuário, operador e cliente com menor número possível de erros e, com isto, obter a minimização dos custos durante todo o ciclo de vida do sistema de controle (MIYAGI, 1996).

Para este trabalho, não serão utilizadas todas as etapas da metodologia apresentada, visto que o SP estudado é relativamente simples. Portanto, as atividades realizadas na execução deste projeto podem ser reorganizadas da seguinte forma:

1. Análise das necessidades;
    a. Identificação do objetivo final do sistema;
    b. Lista preliminar de atuadores;
    c. Lista preliminar de detectores;
2. Levantamento e análise das funções de controle;
    a. Definição do fluxo e das funções de controle;
3. Projeto do Sistema de Controle;
    a. Definição das interfaces e alocação das funções;
    b. Projeto do sistema de controle local;
    c. Projeto do sistema de operação remota.
4. Implementação do Sistema de Controle
    a. Instalação do CLP;
    b. Programação do CLP;
    c. Desenvolvimento do *WebService*;
    d. Desenvolvimento do Web Site.
5. Testes

A elaboração do projeto seguiu-se por etapas. Na sequência do texto, são apresentados os resultados de cada etapa de projeto.

4.2) Identificação do Objetivo Final do Sistema

O sistema de controle para estação de inspeção do Mini CIM deve permitir a monitoração e controle deste sistema localmente e remotamente. O protocolo de comunicação empregado entre o sistema supervisório e o aplicativo local será o *WebService*, desta forma, pretende-se utilizar uma linguagem comum no desenvolvimento de aplicativos web que permita o encapsulamento/modularização das funções, operações e fluxo de dados. Na Figura 10, pode ser visualizado a estrutura do sistema de controle implementado.

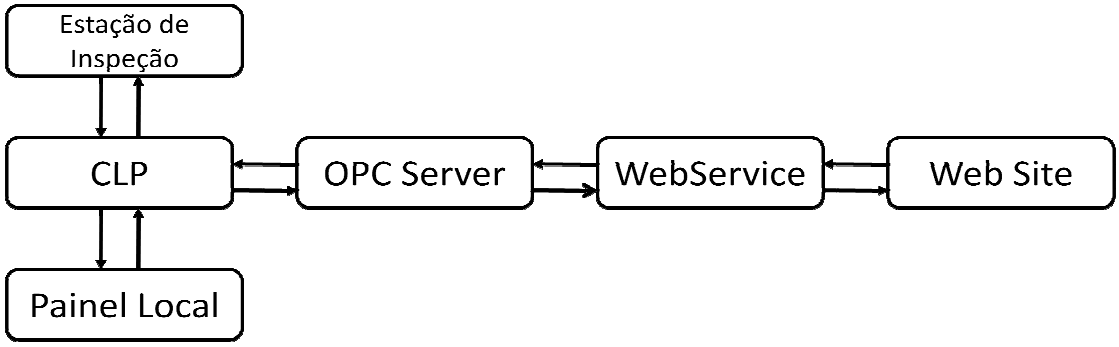


Figura 10 – Estrutura do sistema de controle

Com base nesta estrutura, o desenvolvimento do trabalho foi segmentado conforme será visto ao longo do texto.

4.3) Estação de Inspeção

Trata-se da parte física da estação de inspeção do Mini CIM, incluindo os atuadores (cilindros e válvulas) e sensores. Para esta camada, inicialmente foi obtido o modelo do funcionamento desta estação, com a definição das entradas e saídas das variáveis do sistema e o modelo em PFS/MFG.

4.3.1) Entradas e saídas do sistema

As variáveis de entrada e saídas da estação de inspeção estão listadas nas Tabelas 1 e 2. Também são definidas as respectivas portas do CLP para cada entrada e saída.

Tabela 1 – Lista de Entradas

Cód.	Função	Tipo de dispositivo	Tipo de dado	Pos. mem. CLP
1A	Sobe a peça para inspeção	Cilindro pneumático de dupla ação	--	--
2A	Transporta a peça ao sistema de transporte	Cilindro pneumático de simples ação	--	--
3A	Caracteriza a altura da peça	Cilindro pneumático de simples ação	--	--
1Y1	Desce Plataforma	Solenoide	booleano	I 0.0
1Y2	Sobe Plataforma	Solenoide	booleano	I 0.1
2Y1	Aciona Cilindro Horizontal (Avança)	Solenoide	booleano	I 0.2
3Y1	Aciona Cilindro Vertical (P/ Baixo)	Solenoide	booleano	I 0.3

Tabela 2 – Lista de Saídas

Cód.	Função	Tipo de dispositivo	Tipo de dado	Pos. mem. CLP
B5	Detecta peça metálica	Sensor indutivo	booleano	Q 0.0
B6	Detecta peça na inspeção	Sensor capacitivo	booleano	Q 0.1
B7	Detecta peça não preta	Sensor óptico	booleano	Q 0.2
1B2	Detecta o elevador embaixo	Sensor magnético de proximidade	booleano	Q 0.3
1B2	Detecta o elevador em cima	Sensor magnético de proximidade	booleano	Q 0.4
2B1	Detecta cilindro horizontal recuado	Sensor magnético de proximidade	booleano	Q 0.5
3B1	Detecta cilindro de altura - Posição embaixo	Sensor magnético de proximidade	booleano	Q 0.6
S1	START	Chave de contato	booleano	Q 1.0
S2	RESET	Chave de contato	booleano	Q 1.1
S3	ADJUST	Chave de contato	booleano	Q 1.2
S4	AUTO/MAN	Chave de contato	booleano	Q 1.3
S5	STOP	Chave de contato	booleano	Q 1.4
S6	Comunicação	Chave de contato	booleano	Q 1.6

4.3.2) Levantamento e análise das funções de controle

A função básica da estação de inspeção é determinar se a peça está de acordo com as especificações determinadas pelo usuário. Esta função pode ser subdividida nas seguintes funções secundárias:

- Determinar o tipo de material da peça;
- Determinar a dimensão da peça;
- Movimentar a peça dentro da estação de inspeção.

A estruturação entre a função básica e as funções secundárias pode ser visualizada na Figura 11.

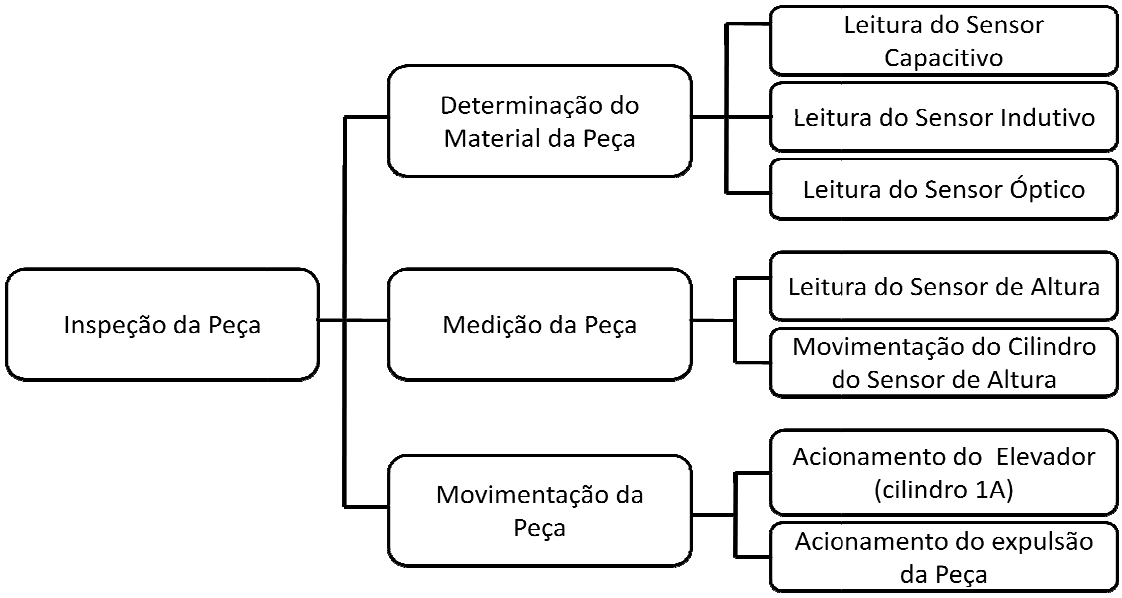


Figura 11 – Estruturação das funções de controle

4.3.3) Definição do fluxo e das funções de controle

Para realização das operações especificadas, deve-se definir os procedimentos que ativam as várias funções de controle anteriormente definidas, isto é, deve-se definir o fluxo das funções de controle.

A utilização do PFS/MFG permite representar os passos em blocos funcionais (atividades) de diferentes níveis conceituais admitindo uma representação estruturada, isto é, os passos descritos em nível conceitual mais alto (PFS) podem ser gradativamente detalhados. A Figura 12 é a representação num nível conceitual macro do fluxo das funções de controle da estação de inspeção do Mini CIM.

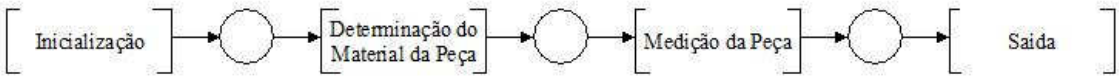


Figura 12 – PFS/MFG das funções de controle (nível macro)

A Figura 13 é a representação em nível mais detalhado da Figura 12, com as especificações das entradas e saídas utilizadas no processo de inspeção.

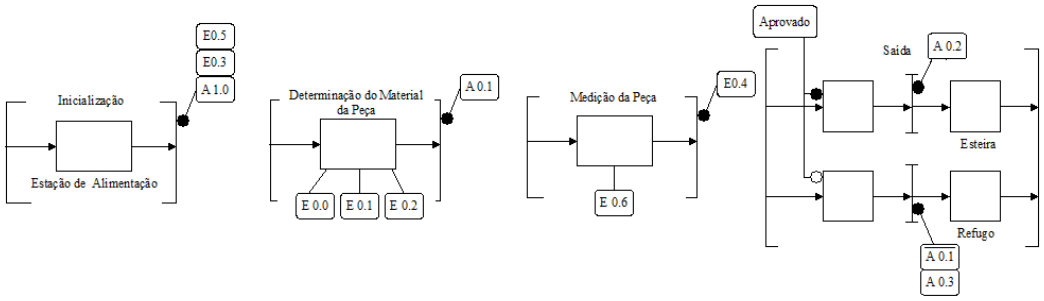


Figura 13 – PFS/MFG das funções de controle (nível de detalhe)

4.4) CLP

Trata-se de controlador programável que atua diretamente com os I/O's da estação de inspeção do Mini CIM. Neste trabalho, o CLP aplicado foi o Siemens WinAC 412, o qual passou pelo processo de instalação e foi programado com base no modelo PFS/MFG da estação de inspeção.

Para ser fazer a comunicação do CLP com o *WebService*, utilizou-se o OPC Server como interface entre a comunicação entre estas duas camadas.

4.4.1) Instalação do CLP

Esta etapa do projeto, consistiu na instalação e configuração do CLP WinAC 412 na estação de inspeção do Mini CIM. Toda a parte de cabos e fios foi refeita, mas mantendo a mesma nomenclatura para a numeração das portas de I/O para o CLP. Nas Figuras 14 e 15, pode-se acompanhar esta etapa do projeto.

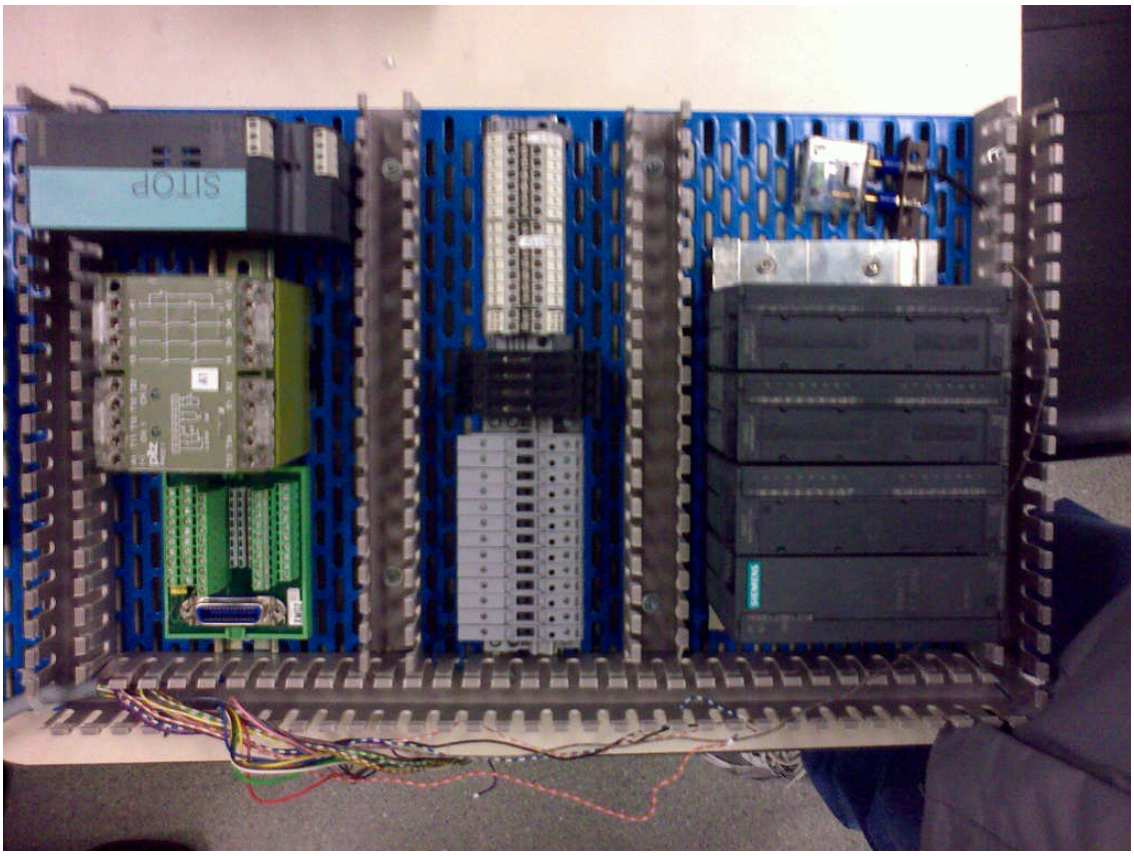


Figura 14 – Painel sem a fiação

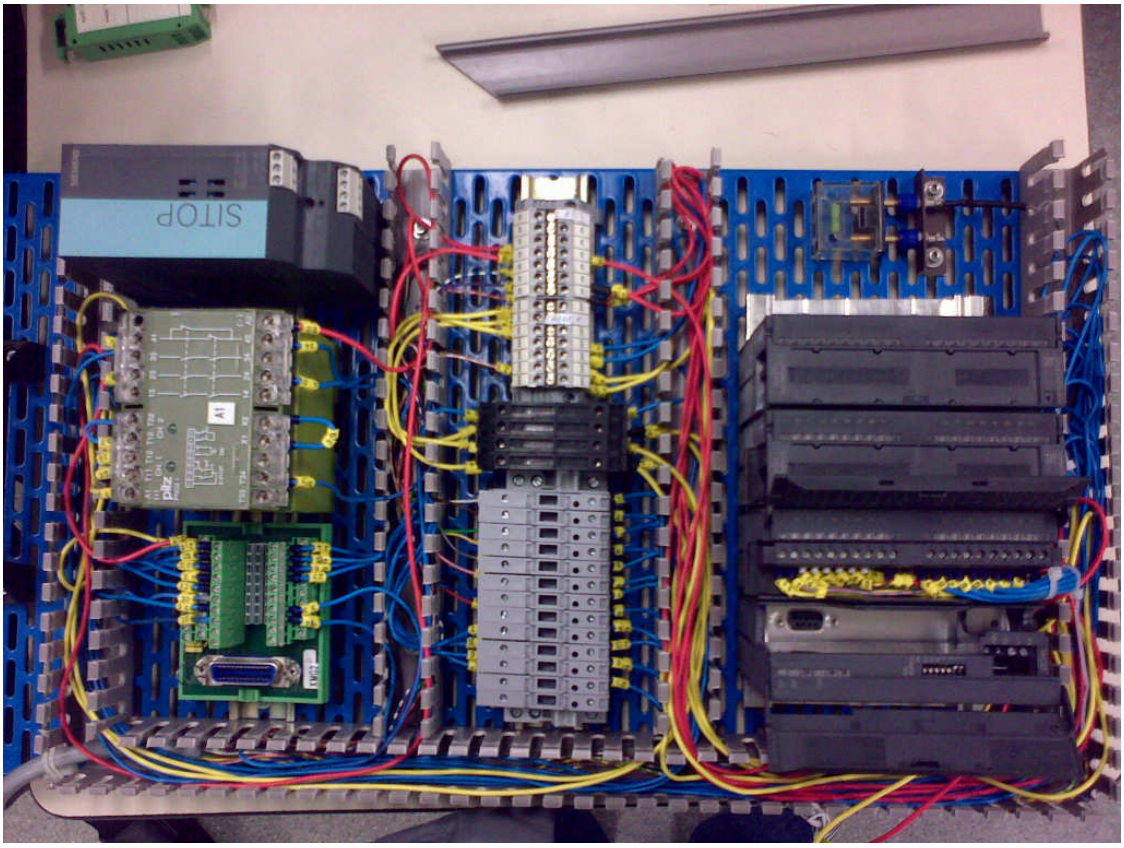


Figura 15 – Painel finalizado

No anexo, pode-se encontrar o diagrama completo da instalação elétrica do CLP.

4.4.2) Programação do CLP

A programação do CLP foi realizada em linguagem *Ladder*, buscando atender aos requisitos levantados no item (4.4.3). Para cada função de controle secundária, determinadas no item (4.4.2), há uma respectiva função no código em *Ladder*, como por exemplo, “Aciona Elevador”.

O programa instalado no CLP está no anexo deste trabalho.

O CLP, com o programa devidamente carregado, permite a operação da estação em dois modos, a operação remota e a operação local. O modo de operação remota permite a monitoração e manipulação das funções de controle do CLP pelo *WebService*.

O *WebService* não terá acesso direto às variáveis de I/O’s do CLP. O programa em *Ladder* permite ao *WebService*, o acesso às variáveis de memória, que por sua vez, executam as funções de controle secundárias. As funções de controle secundário, por sua vez, modificam os valores das I/O’s do CLP. Nas Tabelas 3 e 4, são apresentadas as variáveis de memória de entrada e saída, respectivamente, que são acessadas pelo *WebService*.

Tabela 3 – Variáveis de entrada acessadas pelo WebService

Entradas		
ID	**Nome da Variável**	**Descrição**
W000	mPlataformaSobe	Sobe plataforma
W001	mPlataformaAbaixa	Desce plataforma
W002	mEstendeCilindroPeça	Estende cilindro de expulsão de peça
W003	mRecuaCilindroPeca	Recua cilindro de expulsão de peça
W004	mEstendeCilindroAltura	Estende cilindro do suporte do sensor de altura

4.5) WebService

Trata-se do aplicativo web que deverá se comunicar com o OPC *Server*, permitindo o monitoramento e controle das variáveis do CLP remotamente. Uma das camadas do *WebService* é a responsável por publicar os métodos de monitoração e controle no ambiente da internet.

Tabela 4 – Variáveis de saída acessadas pelo WebService

Saídas		
ID	**Nome da Variável**	**Descrição**
R000	iIndutivo	Indica o valor de leitura do sensor indutivo
R001	iCapacitivo	Indica o valor de leitura do sensor capacitivo
R002	iOtico	Indica o valor de leitura do sensor óptico
R003	iPlataformaEmbaixo	Indica que a plataforma está na posição inferior
R004	iPlataformaAcima	Indica que a plataforma está na posição superior
R005	iCilindroPeca	Indica que o cilindro de expulsão de peça está recuado
R006	iCilindroAltura	Indica que o cilindro do sensor de altura está estendido
R007	mSetup	Indica que a máquina está pronta para inciar a operação
R008	comunInternet	Indica que o modo de operação remoto está habilitado

O WebService foi desenvolvido na linguagem C# (CSharp), devido a sua fácil integração em serviços web, sendo construído com a intenção de disponibilizar na internet um serviço de controle da estação de Inspeção do Mini CIM. Onde está contida toda a lógica de funcionamento da parte de controle remoto (pela internet), ou seja, todo o processamento das ordens recebidas do site são rodadas no *WebService* que encaminha os passos a serem executados para o CLP.

Para a concepção do *WebService* foram utilizados uma biblioteca, que permite a comunicação com o CLP, um arquivo XML, para mapear as variáveis e um arquivo .asmx contendo todos os métodos disponíveis online. O código de todo o projeto está anexo ao trabalho.

4.6) WebSite

Trata-se da interface com usuário que se comunica com o *WebService*, por meio da internet, permitindo a leitura e operação das variáveis de controle da estação de inspeção do Mini CIM.

A estrutura do Web Site foi realizada utilizando-se HTML e CSS. Para a comunicação do Web Site com o *WebService*, foi utilizado Javascript, pois, deste modo permite-se que o site não fosse recarregado a cada método chamado. O *site* serve apenas para fazer a chamada dos métodos, todo o processamento ocorre no *WebService*.

A interface com o usuário pode ser vista na Figura 16.

Figura 16 – Interface com usuário (web site)

Por meio do *web site*, o usuário pode monitorar e controlar a estação de inspeção verificando o estado das saídas do CLP. Uma câmera de vídeo auxilia na monitoração do sistema, permitindo visualizar o funcionamento do sistema.

4.7) Painel de Controle

Trata-se do painel de botões localizados na estação de inspeção do CLP, que possibilita operar a estação localmente. Este painel detém a prioridade no controle da estação em relação modo de operação remota (Figura 17).

Figura 17 – Painel de controle local

Neste painel de controle pode-se optar entre o modo de operação local ou remoto. No caso do modo de operação local, pode-se optar entre o modo de

operação automático ou passo a passo, e ainda pode-se acionar o botão de *start*, parada de emergência, etc.

5) Comentários Finais e Conclusões

O objetivo deste trabalho, o projeto e implementação de um sistema de teleoperação via internet para um sistema produtivo, foi atingido com sucesso. Neste trabalho, o sistema produtivo estudado foi a estação de inspeção da célula de manufatura didática (Mini CIM) do Laboratório de Sistemas de Automação da Escola Politécnica da USP.

O desenvolvimento do sistema de teleoperação foi divido em camadas, conforme a estrutura apresentada na Figura 10. A modularização das partes do projeto apresenta grande vantagem tanto na parte de planejamento e desenvolvimento do projeto, quanto ao desempenho do mesmo. Pode-se citar a utilização do *WebService* como camada que desacopla a interface do usuário (Web Site) com o restante do sistema de controle. Desta maneira, há um encapsulamento das funções de controle no *WebService*, simplificando de maneira significativa o desenvolvimento do Web Site, visto que o desenvolvedor da interface se preocupa unicamente com as interações com o usuário.

A implantação da câmera para monitoração permitiu acompanhar o funcionamento da estação de inspeção sem grandes problemas. Porém, como recomendação pode-se utilizar duas câmeras de 2 Mega Pixels para melhor visualização e eliminação de pontos cegos para o usuário remoto.

O projeto desenvolvido neste trabalho está configurado de maneira que aplica-se unicamente à estação de inspeção do Mini CIM. Contudo, o conceito no qual este fora baseado pode ser aplicado a outros SP. Desta forma, toda a metodologia de projeto pode ser aproveitada para o desenvolvimento de outro sistema de teleoperação de manufatura.

Referências Bibliográficas

ÁLVARES, A.J. e PAULINYI, S.C.A.; "Telerobotics:Through−the−Internet Teleoperation of the ABB IRB 2000 Industrial Robot", Telemanipulator and Telepresence Technologies V − SPIE (The International Society for Optical Enginnering) − Telemanipulator and Telepresence Technologies VI, pp. 259−269, Boston, USA, 1999.

ÁLVARES, A.J. e ROMARIZ, L.J. "TeleRobótica: Metodologia Para o Desenvolvimento de Sistemas Robóticos Teleoperados Via Internet", XV Congresso Brasileiro de Engenharia Mecânica, Águas de Lindóia, SP, 22−26 de Novembro de 1999.

ÁLVARES, A.J. e TOURINO; S.R, "Desenvolvimento de um Robô Móvel Autônomo Teleoperado Via Internet" , Congresso Nacional de Engenharia Mecânica 2000, Natal, RN, 7−12 de agosto de 2000.

ÁLVARES, A.J., e SOUSA, A.I.; "Telemanufatura: Teleoperação Via Web da Máquina de Corte CNC Autocut 2.5", COBEN???????, 2001.

CAMELO, D.M., “Web Service – Curso de especialização de sistemas de informação e aplicações Web”. CESF/FUCAPI. Manaus. 2002.

CASSANDRAS, C.G.; STRICKLAND, S.G. "Sample Path Properties of timed Discreted Event Systems in Discrete Event Dynamic Systems - Analizing Complexity and Performance in the Modern World". New York: IEEE Press, 1992.

CASTRO, E, “Fundamentos da Programação LADDER”, Apostila do Curso Automação da Produção, Universidade Federal de Juiz de Fora, Juiz de Fora, 2007.

CAVALHEIRO, A.C.M.; "Projeto de Sistemas de Controle Modulares e Distribuídos". Dissertação de Mestrado. Escola Politécnica da USP. São Paulo. 2004.

CERAMI, E.; “Web Services Essentials – Distributed applications with XML-RPC, SOAP, UDDI & WSDL”. O’REILLY, 2002.

CHRIST, R.E. R.; FIGUEREDO, M.V.M.; BASSANI, T.; DIAS,J.S.; NIEVOLA, J.C.; “Sistema de monitoração remota de pacientes em tempo-real através da Internet do hospital”. Disponível em: <http://www.sbis.org.br/cbis9/arquivos/603.pdf>. Acesso em Abril de 2008.

GROOVER, M.P., “Automation, Production Systems, and Computer-Integrated Manufacturing”. Prentice Hall. 2000.

GOULART, C.P; “Proposta de um modelo de referência para planejamento e controle da produção em empresas virtuais”. Dissertação de Mestrado. Escola de Engenharia de São Carlos da Universidade de São Paulo. 2000.

INTERNET USAGE WORLD STATISTICS; “The Internet Big Pictute – World Internet Users and Population Stats”. Disponível em:<http://www.internetworldstats.com/stats.htm>. Acesso em junho de 2009.

JUNQUEIRA, F.; “Método para a Modelagem e Simulação Distribuída de Sistemas Produtivos”. Tese de doutorado. Escola Politécnica da USP. São Paulo. 2006

KANEKO, A.M.; "Desenvolvimento de uma interface gráfica para supervisão de um sistema produtivo teleoperado". Trabalho de Formatura. Escola Politécnica da USP. São Paulo. 2008.

KAO, Y.C., e LIN, G.C.; "CAD/CAM Collaboration and remote Machining", Computer Integrated Manufacturing Systems, Vol. 9, No. 3, pp. 149−160, 1996.

KUMAR, R; GARG, V.K.; "Modeling and Control of Logical Discrete Event Systems". Kluwer Academic Publishers. Massachusetts. 1995.

MIYAGI, P.E., “Controle Programável - Fundamentos do Controle de Sistemas a Eventos Discretos”. São Paulo: Editora Edgard Blücher, 1996.

PROFIBUS. Site oficial da organização nacional de usuários de comunicação Profibus, 2006. Disponível em: <www.profibus.org.br>. Acesso em Maio de 2009.

ROCHA, L.F.; CSO e a importância da monitoração remota. 11 de agosto de 2003. Disponível em: <http://www.cert.br/docs/reportagens/2003/2003-08-11.html>. Acesso em Fevereiro de 2006.

SANTOS FILHO, D.J.; Aspectos de projeto do controle de sistemas produtivos. 116 p. Tese (Livre Docência) – Escola Politécnica da USP. São Paulo, 2000.

SIEMENS; “Basic of PLCs”.2004. Disponível em: <http://www.scribd.com/doc/7261014/Siemens-Basics-of-Plc>. Acesso em: Maio de 2009.

SILVA FILHO, B.S.; “Curso de controladores lógico programáveis”. Laboratório de Engenharia Elétrica. Faculdade de Engenharia. Universidade do Estado do Rio de Janeiro. Rio de Janeiro. 1999.

SOUZA, E.M.; Desenvolvimento de componentes para facilitar a monitoração remota de redes de computadores usando a ferramenta WebManager. 2002. Disponível em: <http://www.dsc.ufcg.edu.br/~copin/pesquisa/bancodissertacoes/2002/xsonMachadoSouza.pdf>. Acesso em Abril de 2008.

STEFFEN, M.; ZANINI, L.F.A.; "Ferramenta Computacional para Designação de Controladores Programáveis em Arquiteturas de Controle Distribuídas". Trabalho de Formatura. Escola Politécnica da USP. São Paulo. 2005.

TAYLOR, K. & TREVELYAN, J.; "A Telerobot on the World Wide Web", National Conference of the Australian Robot Association, Melbourne, 1995. Disponível em: <http://telerobot.mech.uwa.edu.au>. Acesso em: 5 de Julho de 2007.

TOURINO, S.G; "Guiagem de Robô Móvel XR4000 para Inspeção de Tubulações Industriais Soldadas Via Internet", Projeto de Conclusão de Curso, GRACO − Grupo de Automação e Controle, UnB, Brasília, 2000.

MUSEU VIRTUAL DA INFORMATICA;; "Breve História da Internet", UNIVERSIDADE DO MINHO. Disponível em: <http://piano.dsi.uminho.pt/museuv/INTERNET.PDF>. Acesso em: 14 de junho de 2009.

VILLANI, E.; “Modelagem e análise de sistemas supervisórios híbridos”. Tese de doutorado. Escola Politécnica da Universidade de São Paulo, 2003.

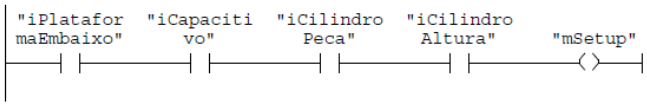
W3SCHOOLS.COM; “Web Services Tutorial”, Disponível em: <www.w3schools.com/webservices/default.asp>. Acesso em: 28 de abril de 2009.

ZHAI, S., MILGRAM, P.; "A Telerobotic Virtual Control System", Proc. SPIE. Boston, 1991.

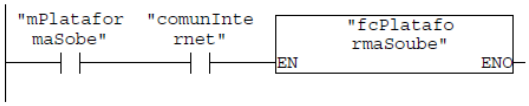
Apêndice A - Programa do CLP

Block: OB1 "Main Program Sweep (Cycle)"

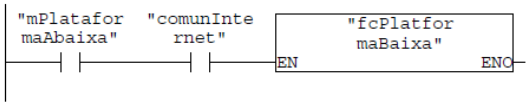
Network: 1 Detecta se a estacao esta na posição inicial



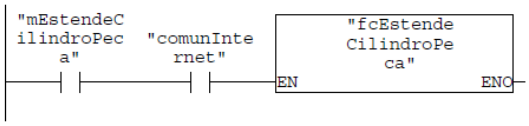
Network: 9



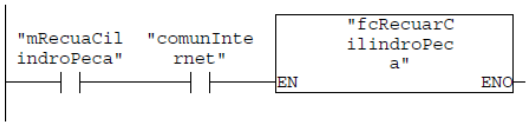
Network: 10



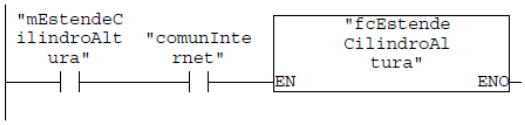
Network: 11 Tira a peca da plataforma de identificacao da peca



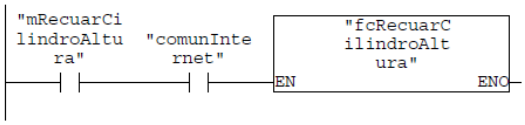
Network: 12



Network: 13



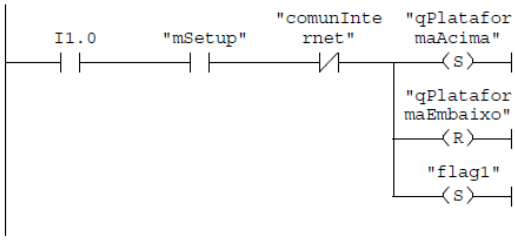
Network: 14



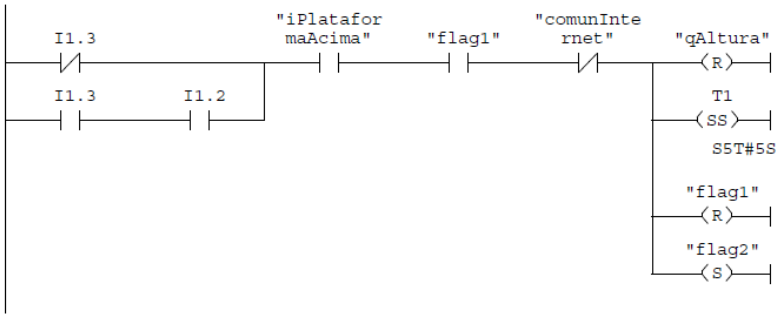
Network: 15 Verificação de comunicação

verifica se a comunicação é local ou via internet

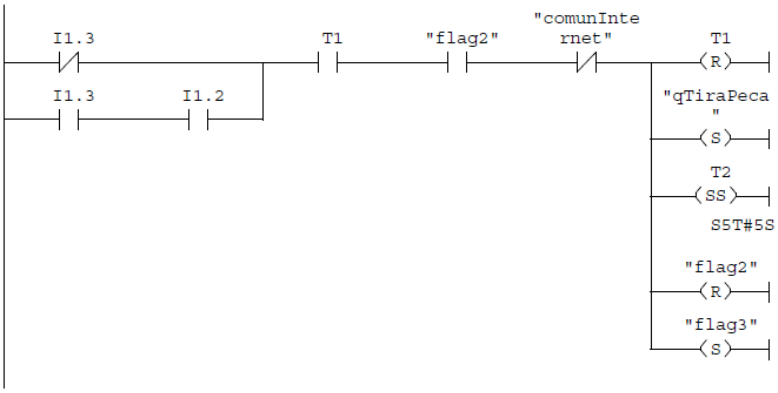
Network: 16 Soube a plataformade identificação da peca



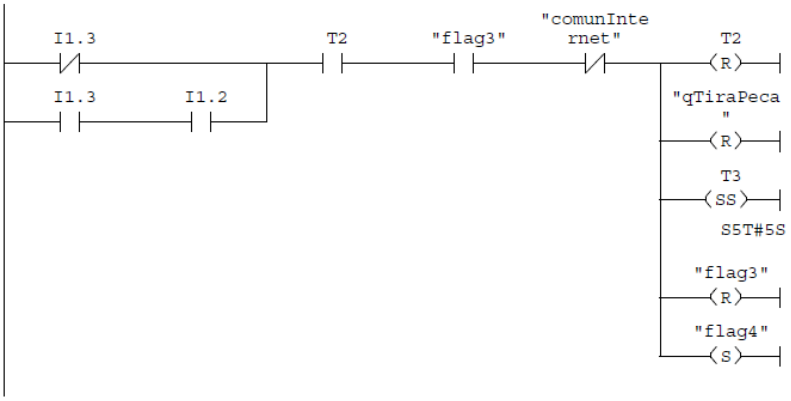
Network: 17 Tira a peca da plataforma de identificacao da peca



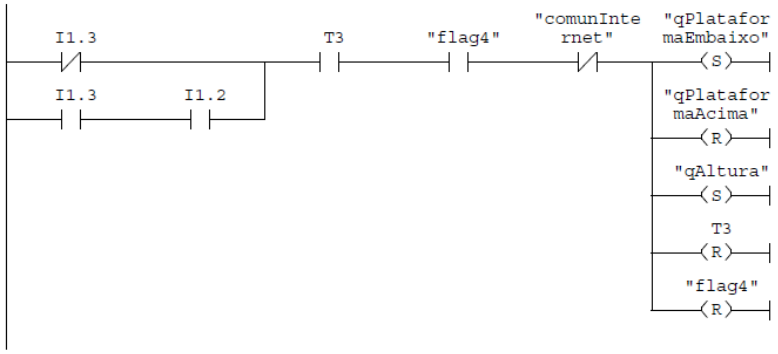
Network: 18



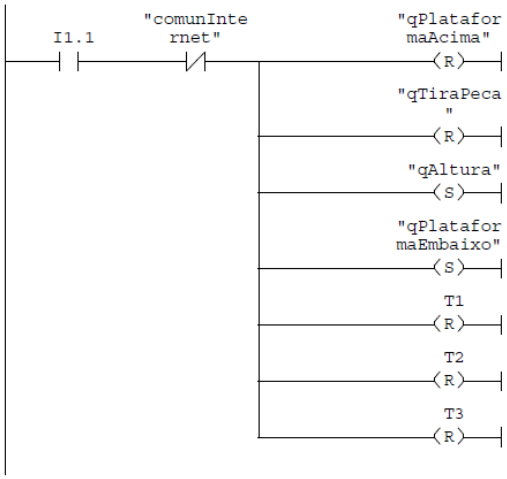
Network: 19



Network: 20 Baixa a plataformade identificação da peca

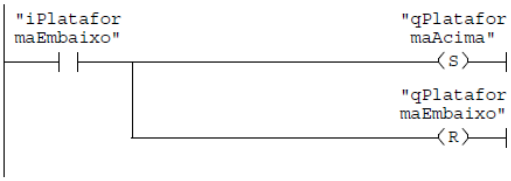


Network: 21 Reset



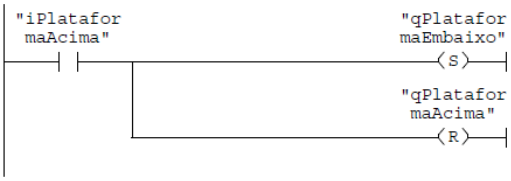
Block: FC1

Network: 1 Soube a plataformade identificação da peca



Block: FC2

Network: 1



Block: FC3

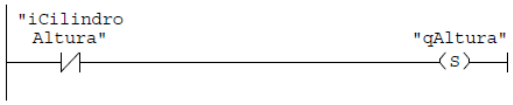
Network: 1 Tira a peca da plataforma de identificacao da peca

Block: FC4

Network: 1

Block: FC5

Network: 1



Block: FC6

Network: 1

Apêndice B - Código Fonte do WebService

WebServ.asmx.cs

```
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Linq;
using System.Web;
using System.Web.Services;
using System.Web.Services.Protocols;
using System.Xml.Linq;
using System.Windows.Forms;
using System.Xml;

namespace Service
{

    [WebService(Namespace = "http://tempuri.org/")]
    [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
    [System.Web.Script.Services.ScriptService]
    public class WebServ : System.Web.Services.WebService
    {

        private XmlDocument xmlDoc;
        private opcconn.opcClient opc;

        [WebMethod]
        public void CarregaXml()
        {
            xmlDoc = new XmlDocument();
            Application.Lock();
            xmlDoc.Load("http://192.168.0.197/Inspecao/xml/opcconfig.xml");
            Application.Contents["arquivoXML"] = xmlDoc;
            Application.UnLock();
            //MessageBox.Show("XML carregado");

        }

        [WebMethod]
        public void ConnOpc()
        {
            if (Application.Contents["arquivoXML"] == null)
            {
                //MessageBox.Show("Arquivo XML não encontrado");
            }
            else
            {
                xmlDoc = new XmlDocument();
                xmlDoc = (XmlDocument)Application.Contents["arquivoXML"];
                XmlNode xmlServer = xmlDoc.GetElementsByTagName("server")[0];
                opc = new opcconn.opcClient(xmlServer.Attributes.GetNamedItem("name").Value);
                if (opc.ErrorMessage.Length > 0)
                {
                    opc.Dispose();
                }
```

```
            else
            {
                foreach (XmlNode xmlGroup in xmlServer.SelectNodes("group"))
                {
                    opc.AddGroup(xmlGroup.Attributes.GetNamedItem("name").Value);
                    if (opc.ErrorMessage.Length == 0)
                    {
                        foreach (XmlNode xmlItem in xmlGroup.SelectNodes("item"))
                        {

opc.GetGroupByPosition(0).AddItem(xmlItem.Attributes.GetNamedItem("id").Value,
xmlItem.Attributes.GetNamedItem("memory").Value);
                            if (opc.GetGroupByPosition(0).ErrorMessage.Length > 0)
                            {
                                //MessageBox.Show(opc.GetGroupByPosition(0).ErrorMessage);
                            }
                        }
                    }
                    else
                    {
                        //MessageBox.Show(opc.ErrorMessage);
                    }
                }
                Application.Lock();
                Application.Contents["ConexaoOPC"] = opc;
                Application.UnLock();
                //MessageBox.Show("Conexão OPC montada");
            }
        }
    }

    [WebMethod]
    public void automatico(string preto,string rosa, string prata, double dimensions)
    {

        setStart();
        System.Threading.Thread.Sleep(4000);

        string capacitivo, indutivo, optico;
        string lpreto,lprata,lrosa;

        lpreto = "false";
        lrosa = "false";
        lprata = "false";

        //Checa mSetup

        string mSetup = lerPorta("R007");
        if (mSetup == "0")
        {
            //MessageBox.Show("Erro de inicialização.");
        }
        else
        {
            //Inicio do programa

            //Funcao detectaMaterial()

            capacitivo = lerPorta("R001");
```

```
            indutivo = lerPorta("R000");
            optico = lerPorta("R002");

            if (capacitivo == "0")
            {
                //MessageBox.Show("Não há peça na plataforma");
            }
            else
            {
                if (optico == "0" && indutivo == "0")
                {
                    lpreto = "true";
                    //return cor
                }
                else
                {
                    if (indutivo == "0")
                    {
                        lrosa = "true";
                        //return cor
                    }
                    else
                    {
                        lprata = "true";
                        //return cor
                    }
                }
            }
            //fim detectaMaterial()

            if (((lpreto == "true") && (preto == "true")) || ((lrosa == "true") && (rosa == "true")) ||
((lprata == "true") && (prata == "true")))
            {

                sobePlataforma();

                while (lerPorta("R004") == "0") { }

                //funçao medeAltura()

                Boolean decisao = true;

                if (decisao == false)
                {

                    descePlataforma();
                    avancaPeca();

                    System.Threading.Thread.Sleep(2000);

                    recuaPeca();
                }
                else
                {

                    sobeAltura();
                    System.Threading.Thread.Sleep(2000);
                    avancaPeca();
                    System.Threading.Thread.Sleep(2000);
```

```
                recuaPeca();
                System.Threading.Thread.Sleep(1000);
                descePlataforma();
                desceAltura();

            }
        }
        else
            avancaPeca();

            System.Threading.Thread.Sleep(2000);

            recuaPeca();

            MessageBox.Show("Peça dispensada.");
        {




        }

    }

}

[WebMethod]
public void avancaPeca()
{
    if (Application.Contents["ConexaoOPC"] == null)
    {
        //MessageBox.Show("Conexão com o OPC Server não encontrada");
    }
    else
    {
        opc = new opcconn.opcClient();
        opc = (opcconn.opcClient)Application.Contents["ConexaoOPC"];
        opcconn.opcClientGroup group = opc.GetGroupByPosition(0);
        if ((lerPorta("R004") == "-1") && lerPorta("R006") == "-1")
        {
            //MessageBox.Show("Operação arriscada, suba primeiro o medidor de altura");
        }
        else {
            group.GetItemById("W003").Write("0");
            group.GetItemById("W002").Write("1");
        }
        group = null;
    }
}
[WebMethod]
public void recuaPeca()
{
    if (Application.Contents["ConexaoOPC"] == null)
    {
        //MessageBox.Show("Conexão com o OPC Server não encontrada");
    }
    else
    {
        opc = new opcconn.opcClient();
```

```
        opc = (opcconn.opcClient)Application.Contents["ConexaoOPC"];
        opcconn.opcClientGroup group = opc.GetGroupByPosition(0);
        group.GetItemById("W002").Write("0");
        group.GetItemById("W003").Write("1");
        group = null;
    }
}

[WebMethod]
public void sobePlataforma()
{
    if (Application.Contents["ConexaoOPC"] == null)
    {
        //MessageBox.Show("Conexão com o OPC Server não encontrada");
    }
    else
    {
        opc = new opcconn.opcClient();
        opc = (opcconn.opcClient)Application.Contents["ConexaoOPC"];
        opcconn.opcClientGroup group = opc.GetGroupByPosition(0);
        if(lerPorta("R005")=="0"){
            //MessageBox.Show("Operação arriscada, recue primeiro o cilindro peça.");
        }
        else{
            group.GetItemById("W001").Write("0");
            group.GetItemById("W000").Write("1");
        }
        group = null;
    }
}
[WebMethod]
public void descePlataforma()
{
    if (Application.Contents["ConexaoOPC"] == null)
    {
        //MessageBox.Show("Conexão com o OPC Server não encontrada");
    }
    else
    {
        opc = new opcconn.opcClient();
        opc = (opcconn.opcClient)Application.Contents["ConexaoOPC"];
        opcconn.opcClientGroup group = opc.GetGroupByPosition(0);
        if (lerPorta("R005") == "0")
        {
            //MessageBox.Show("Operação arriscada, recue primeiro o cilindro peça.");
        }
        else
        {
            group.GetItemById("W000").Write("0");
            group.GetItemById("W001").Write("1");
        }
        group = null;
    }
}

[WebMethod]
public void sobeAltura()
{
    if (Application.Contents["ConexaoOPC"] == null)
```

```
    {
        //MessageBox.Show("Conexão com o OPC Server não encontrada");
    }
    else
    {
        opc = new opcconn.opcClient();
        opc = (opcconn.opcClient)Application.Contents["ConexaoOPC"];
        opcconn.opcClientGroup group = opc.GetGroupByPosition(0);
        group.GetItemById("W004").Write("0");
        group.GetItemById("W005").Write("1");
        group = null;
    }
}
[WebMethod]
public void desceAltura()
{
    if (Application.Contents["ConexaoOPC"] == null)
    {
        //MessageBox.Show("Conexão com o OPC Server não encontrada");
    }
    else
    {
        opc = new opcconn.opcClient();
        opc = (opcconn.opcClient)Application.Contents["ConexaoOPC"];
        opcconn.opcClientGroup group = opc.GetGroupByPosition(0);
        if((lerPorta("R004")=="-1")&&(lerPorta("R005")=="0")){
            //MessageBox.Show("Operação arriscada, recue primeiro o cilindro da peça.");
        }
        else{
            group.GetItemById("W005").Write("0");
            group.GetItemById("W004").Write("1");
        }group = null;
    }
}

public string lerPorta(string var)
{
    opc = new opcconn.opcClient();
    opc = (opcconn.opcClient)Application.Contents["ConexaoOPC"];
    opcconn.opcClientGroup group = opc.GetGroupByPosition(0);
    string variavel = group.GetItemById(var).Read();
    group = null;
    return variavel;
}

[WebMethod]
public void setStart()
{
    if (Application.Contents["ConexaoOPC"] == null)
    {
        //MessageBox.Show("Conexão com o OPC Server não encontrada111");
    }
    else
    {
        opc = new opcconn.opcClient();
        opc = (opcconn.opcClient)Application.Contents["ConexaoOPC"];
        opcconn.opcClientGroup group = opc.GetGroupByPosition(0);
        group.GetItemById("W000").Write("0");
        group.GetItemById("W001").Write("0");
```

```
        group.GetItemById("W002").Write("0");
        group.GetItemById("W003").Write("0");
        group.GetItemById("W004").Write("0");
        group.GetItemById("W005").Write("0");

        sobeAltura();
        System.Threading.Thread.Sleep(2000);
        recuaPeca();
        System.Threading.Thread.Sleep(2000);
        descePlataforma();
        desceAltura();
        group = null;
    }
}

[WebMethod]
public string leituraComunWEB(string porta)
{
    string result = "";
    if (Application.Contents["ConexaoOPC"] != null)
    {
        result = lerPorta(porta);
    }
    return result;
}

[WebMethod]
public string leituraSensor(string porta)
{
    string result = "";
    if (Application.Contents["ConexaoOPC"] != null)
    {
        result = lerPorta(porta);
    }
    return result;
}

[WebMethod]
public string corPeca() {

    string Cor="npeca";
    string capacitivo, indutivo, optico;
    capacitivo = lerPorta("R001");
    indutivo = lerPorta("R000");
    optico = lerPorta("R002");


    if(capacitivo=="-1"){
        Cor = "Preto";
        if(optico=="-1"){
            Cor = "Rosa";
            if(indutivo=="-1"){
                Cor = "Prata";
            }
        }
    }
    if (lerPorta("R003") == "0")
    {
        Cor = "mantercor";
```

```
            }

            return Cor;
        }
    }

}
```

Opcconn.cs

```
using System;
using System.Collections.Generic;
using System.Runtime.InteropServices;
using System.Text;
using OpcRcw.Da;
using System.Windows.Forms;

namespace opcconn {

    public class opcconn
    {

        public void ola() {
            MessageBox.Show("olaaa");
        }

    }

    public class opcClient {
        #region Private variables
        private Guid _iIdRequiredInterface = typeof(IOPCItemMgt).GUID;
        private IOPCServer _pIOPCServer;
        private string _errorMessage = "";
        private int _localId = 0x409;
        private List<opcClientGroup> _listOfGroups = new List<opcClientGroup>();
        #endregion

        #region Methods
        public opcClient(string strServerName) {
            Type svrComponentType = System.Type.GetTypeFromProgID(strServerName);

            _errorMessage = "";
            try {
                _pIOPCServer = (IOPCServer)System.Activator.CreateInstance(svrComponentType);
            }
            catch (System.Exception error) {
                _errorMessage = String.Format("Erro ao criar o objeto servidor:-{0}", error.Message);
            }
        }
        public void Dispose() {
            _errorMessage = "";
            try {
                foreach (opcClientGroup group in _listOfGroups) {
                    group.Dispose();
                }
                if (_pIOPCServer != null)
                    Marshal.ReleaseComObject(_pIOPCServer);
            }
            catch (System.Exception error) {
```

```
            _errorMessage = error.Message;
        }
    }
    public void AddGroup(string strGroupName) {
        _errorMessage = "";
        try {
            opcClientGroup group = new opcClientGroup(_pIOPCServer, strGroupName,
_localId, _iIdRequiredInterface);
            if (group.ErrorMessage.Length == 0) {
                _listOfGroups.Add(group);
            }
            else {
                _errorMessage = group.ErrorMessage;
                group.Dispose();
            }
        }
        catch (System.Exception error) {
            _errorMessage = error.Message;
        }
    }
    public opcClientGroup GetGroupByName(string strGroupName) {
        foreach (opcClientGroup group in _listOfGroups) {
            if (String.Compare(strGroupName, group.Name) == 0)
                return group;
        }
        _errorMessage = "Nenhum grupo foi encontrado";
        return null;
    }
    public opcClientGroup GetGroupByPosition(int intGroupPosition) {
        if ((intGroupPosition >= 0) && (intGroupPosition < _listOfGroups.Count)) {
            return _listOfGroups[intGroupPosition];
        }
        else {
            _errorMessage = "O índice excede o número de grupos cadastrados";
            return null;
        }
    }
    #endregion

    #region Properties
    public string ErrorMessage {
        get {
            return _errorMessage;
        }
        private set{
        }
    }
    public int GroupCount {
        get {
            return _listOfGroups.Count;
        }
        private set {
        }
    }
    public int LocalId {
        get {
            return _localId;
        }
        set {
```

```
            _localId = value;
        }
    }
    #endregion
}
public class opcClientGroup {
    #region Private variables
    private string _strGroupName = "";
    private Object _pObjGroup;
    private IOPCSyncIO _pIOPCSyncIO;
    private IOPCServer _pIOPCServer;
    private int _localId;
    private int _nSvrGroupID;
    private string _errorMessage = "";
    private List<opcClientGroupItem> _listOfItems = new List<opcClientGroupItem>();
    #endregion

    #region Methods
    public opcClientGroup(IOPCServer pIOPCServer, string strGroupName, int localId, Guid
iIdRequiredInterface) {
        // Initialise Group properties
        int bActive = 0;
        int dwRequestedUpdateRate = 250;
        int hClientGroup = 0;
        int pRevUpdateRate;
        int TimeBias = 0;
        float deadband = 0;
        // Access unmanaged COM memory
        GCHandle hTimeBias = GCHandle.Alloc(TimeBias, GCHandleType.Pinned);
        GCHandle hDeadband = GCHandle.Alloc(deadband, GCHandleType.Pinned);

        _errorMessage = "";
        try {
            /* 2. Add a new group
                Add a group object and query for interface IOPCItemMgt
                Parameter as following:
                [in] not active, so no OnDataChange callback
                [in] Request this Update Rate from Server
                [in] Client Handle, not necessary in this sample
                [in] No time interval to system UTC time
                [in] No Deadband, so all data changes are reported
                [in] Server uses english language to for text values
                [out] Server handle to identify this group in later calls
                [out] The answer from Server to the requested Update Rate
                [in] requested interface type of the group object
                [out] pointer to the requested interface
            */

            _pIOPCServer = pIOPCServer;
            _strGroupName = strGroupName;
            _localId = localId;

            pIOPCServer.AddGroup(_strGroupName,
                bActive,
                dwRequestedUpdateRate,
                hClientGroup,
                hTimeBias.AddrOfPinnedObject(),
                hDeadband.AddrOfPinnedObject(),
                localId,
```

```
                out _nSvrGroupID,
                out pRevUpdateRate,
                ref iIdRequiredInterface,
                out _pObjGroup);

            // Get the IOPOCSyncIO interface pointer from the group pointer.
            // pIOPCSyncIO also needs to be released when not in use.
            _pIOPCSyncIO = (IOPCSyncIO)_pObjGroup;
        }
        catch (System.Exception error) {
            _errorMessage = String.Format("Erro ao criar um objeto grupo:-{0}", error.Message);
        }
        finally {
            if (hDeadband.IsAllocated)
                hDeadband.Free();
            if (hTimeBias.IsAllocated)
                hTimeBias.Free();
        }
    }
    public void Dispose() {
        _errorMessage = "";
        try {
            if (_pIOPCSyncIO != null)
                Marshal.ReleaseComObject(_pIOPCSyncIO);
            if (_pObjGroup != null) {
                _pIOPCServer.RemoveGroup(_nSvrGroupID, 0);
                Marshal.ReleaseComObject(_pObjGroup);
            }
        }
        catch (System.Exception error) {
            _errorMessage = error.Message;
        }
    }
    public void AddItem(string strId, string strItemName) {
        _errorMessage = "";
        try {
            opcClientGroupItem item = new opcClientGroupItem(strId, strItemName, _pObjGroup,
_pIOPCServer, _pIOPCSyncIO, _localId);
            if (item.ErrorMessage.Length == 0) {
                _listOfItems.Add(item);
            }
            else {
                _errorMessage = item.ErrorMessage;
            }
        }
        catch (System.Exception error) {
            _errorMessage = error.Message;
        }
    }
    public opcClientGroupItem GetItemByName(string strItemName) {
        foreach (opcClientGroupItem item in _listOfItems) {
            if (String.Compare(strItemName, item.Name) == 0)
                return item;
        }
        _errorMessage = "Nenhum item foi encontrado";
        return null;
    }
    public opcClientGroupItem GetItemById(string strId) {
        foreach (opcClientGroupItem item in _listOfItems) {
```

```
            if (String.Compare(strId, item.Id) == 0)
                return item;
        }
        _errorMessage = "Nenhum item foi encontrado";
        return null;
    }
    public opcClientGroupItem GetItemByPosition(int intItemPosition) {
        if ((intItemPosition >= 0) && (intItemPosition < _listOfItems.Count)) {
            return _listOfItems[intItemPosition];
        }
        else {
            _errorMessage = "O índice excede o número de itens cadastrados no grupo";
            return null;
        }
    }
    #endregion

    #region Properties
    public string ErrorMessage {
        get {
            return _errorMessage;
        }
        private set {
        }
    }
    public int ItemCount {
        get {
            return _listOfItems.Count;
        }
        private set {
        }
    }
    public string Name {
        get {
            return _strGroupName;
        }
        private set {
        }
    }
    #endregion
}
public class opcClientGroupItem {
    #region Private variables
    private string _strItemName = "";
    private string _strId = "";
    private Object _pObjGroup;
    private IOPCServer _pIOPCServer;
    private IOPCSyncIO _pIOPCSyncIO;
    private int _localId;
    private string _errorMessage = "";
    private int[] nItemSvrID;
    #endregion

    #region Methods
    public opcClientGroupItem(string strId, string strItemName, Object pObjGroup,
IOPCServer pIOPCServer, IOPCSyncIO pIOPCSyncIO, int localId) {
        _errorMessage = "";
        _strId = strId;
        _strItemName = strItemName;
```

```
            _pObjGroup = pObjGroup;
            _pIOPCServer = pIOPCServer;
            _pIOPCSyncIO = pIOPCSyncIO;
            _localId = localId;

            /* 3. Add items to the group */
            OPCITEMDEF[] ItemDefArray = new OPCITEMDEF[1];

            ItemDefArray[0].szAccessPath = "";       // Accesspath not needed for this sample
            ItemDefArray[0].szItemID = _strItemName; // ItemID, see above
            ItemDefArray[0].bActive = 1;             // item is active
            ItemDefArray[0].hClient = 1;             // client handle
            ItemDefArray[0].dwBlobSize = 0;          // blob size
            ItemDefArray[0].pBlob = IntPtr.Zero;     // pointer to blob
            ItemDefArray[0].vtRequestedDataType = 2; // return values in native (cannonical)
datatype
            //ItemDefArray[0].wReserved = 0;          // Este achei no exemplo do site "http://lhcb-
online.web.cern.ch/lhcb-online/ecs/opcevaluation/opcclienttutorial/SimpleClient.html"

            // initialize output parameters.
            IntPtr pResults = IntPtr.Zero;
            IntPtr pErrors = IntPtr.Zero;
            try {
                // Add items to group
                ((IOPCItemMgt)_pObjGroup).AddItems(1, ItemDefArray, out pResults, out pErrors);
                // Unmarshal to get the server handles out fom the m_pItemResult after checking the
errors
                int[] errors = new int[1];
                Marshal.Copy(pErrors, errors, 0, 1);
                if (errors[0] == 0) {
                    OPCITEMRESULT result = (OPCITEMRESULT)Marshal.PtrToStructure(pResults,
typeof(OPCITEMRESULT));
                    // Allocate integer array
                    nItemSvrID = new int[1];
                    nItemSvrID[0] = result.hServer;
                }
                else {
                    String pstrError;
                    _pIOPCServer.GetErrorString(errors[0], localId, out pstrError);
                    _errorMessage = pstrError + "; strItemName = " + _strItemName;
                }
                // Free indirect structure elements
                Marshal.DestroyStructure(pResults, typeof(OPCITEMRESULT));
            }
            catch (System.Exception error) {
                _errorMessage = error.Message;
            }
            finally {
                // Free the unmanaged COM memory
                if (pResults != IntPtr.Zero) {
                    Marshal.FreeCoTaskMem(pResults);
                    pResults = IntPtr.Zero;
                }
                if (pErrors != IntPtr.Zero) {
                    Marshal.FreeCoTaskMem(pErrors);
                    pErrors = IntPtr.Zero;
                }
            }
        }
```

```
        public string Read() {
            string strRetorno = "";

            // Access unmanaged COM memory
            IntPtr pItemValues = IntPtr.Zero;
            IntPtr pErrors = IntPtr.Zero;

            try {
                // Sync read from device
                _pIOPCSyncIO.Read(OPCDATASOURCE.OPC_DS_DEVICE, 1, nItemSvrID, out
pItemValues, out pErrors);
                // Unmarshal the returned memory to get the item state out fom the ppItemValues
                // after checking errors
                int[] errors = new int[1];
                Marshal.Copy(pErrors, errors, 0, 1);
                if (errors[0] == 0) {
                    OPCITEMSTATE pItemState =
(OPCITEMSTATE)Marshal.PtrToStructure(pItemValues, typeof(OPCITEMSTATE));
                    // update the UI
                    strRetorno = String.Format("{0}", pItemState.vDataValue);
                    // Free indirect variant element, other indirect elements are freed by
Marshal.DestroyStructure(...)
                    DUMMY_VARIANT.VariantClear((IntPtr)((int)pItemValues + 16));
                }
                else {
                    String pstrError;
                    _pIOPCServer.GetErrorString(errors[0], _localId, out pstrError);
                    _errorMessage = pstrError;
                }

                // Free indirect structure elements
                Marshal.DestroyStructure(pItemValues, typeof(OPCITEMSTATE));
            }
            catch (System.Exception error) {
                _errorMessage = error.Message;
            }
            finally {
                // Free the unmanaged COM memory
                if (pItemValues != IntPtr.Zero) {
                    Marshal.FreeCoTaskMem(pItemValues);
                    pItemValues = IntPtr.Zero;
                }
                if (pErrors != IntPtr.Zero) {
                    Marshal.FreeCoTaskMem(pErrors);
                    pErrors = IntPtr.Zero;
                }
            }
            return strRetorno;
        }
        public void Write(string strValue) {
            // Access unmanaged COM memory
            IntPtr pErrors = IntPtr.Zero;

            object[] values = new object[1];
            values[0] = strValue;

            try {
                _pIOPCSyncIO.Write(1, nItemSvrID, values, out pErrors);
                int[] errors = new int[1];
```

```
                Marshal.Copy(pErrors, errors, 0, 1);

                String pstrError;
                _pIOPCServer.GetErrorString(errors[0], _localId, out pstrError);
                _errorMessage = pstrError;
            }
            catch (System.Exception error) {
                _errorMessage = error.Message;
            }
            finally {
                // Free the unmanaged COM memory
                if (pErrors != IntPtr.Zero) {
                    Marshal.FreeCoTaskMem(pErrors);
                    pErrors = IntPtr.Zero;
                }
            }
        }
        #endregion

        #region Properties
        public string ErrorMessage {
            get {
                return _errorMessage;
            }
            private set {
            }
        }
        public string Name {
            get {
                return _strItemName;
            }
            private set {
            }
        }
        public string Id {
            get {
                return _strId;
            }
            private set {
            }
        }
        #endregion
    }
    [ComVisible(true), StructLayout(LayoutKind.Sequential, Pack = 2)]
    public class DUMMY_VARIANT {
        [DllImport("oleaut32.dll")]
        public static extern int VariantClear(IntPtr addrofvariant);
    }
}
```

Apêndice C - Código Fonte do Web Site

Default.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="Service._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
   <title>MiniCIM - Estação de Inspeção</title>

   <script language="javascript" type="text/javascript" src="codigoJS.js">

   </script>
   <style type="text/css" >

       .bot{width: 200px;height: 31px;}
       .caixabot{width:210px; float:left; margin-left:23px; margin-top:100px;}
       .inputs{float:left; width:379px; height:109px;}
       .meio{float:left; width:379px; margin-top:90px;margin-left:27px;}
       .direita{float:left;margin-top:90px;}
       .style1{height: 35px;}
       .style2{border-style: inset; border-width: medium; height:30px; width: 222px; background-
color: #003399; margin-left:16px;}
       .style3{height:30px; float:left; font-weight: bold; color: #FFFFFF;margin-top:4px; margin-
left:2px;}
       .style4{margin-left:2px; margin-right:5px; width:20px; height:20px;float:right; margin-
top:4px; background-color:#FF0000;}
       .style5{width:379px; height:290px; background-color:White; float:left; }
       .style6{margin-left:2px;margin-right:5px; width:20px; height:20px;float:right; margin-top:4px;
background-color:#FFFF00;}

   </style>

</head>
<body onload="setInterval('atualizacao()', 2000);" >
<div style=" width:100%;" align="center">
   <form id="form1" runat="server" style="background-repeat: no-repeat; width:900px;
height:500px; background-image:url(imagens/fundo.jpg);">
   <div>
      <asp:ScriptManager ID="ScriptManager1" runat="server" >
         <Services>
            <asp:ServiceReference Path="WebServ.asmx" InlineScript="False" />
         </Services>
      </asp:ScriptManager>
   <div>

         <div class="caixabot">
            <table >
               <tr>
                  <td class="style1"><input class="bot" id="CarregaXML" type="button"
value="Carrega XML"
                        onclick="return CarregaXML_onclick()" /></td>
               </tr>
               <tr>
```

```
                <td class="style1"><input class="bot" id="ConectaOPC" type="button"
value="Conecta OPC"
                    onclick="return ConectaOPC_onclick()" /></td>
            </tr>
            <tr>
                <td class="style1"><input class="bot" id="ExpulsaPeca" type="button"
value="Avançar Cilindro Peça"
                    onclick="return ExpulsaPeca_onclick()" /></td>
            </tr>
            <tr>
                <td class="style1"><input class="bot" id="RecuaPeca" type="button"
value="Recuar Cilindro Peça"
                    onclick="return RecuaPeca_onclick()" /></td>
            </tr>
            <tr>
                <td class="style1"><input class="bot" id="SobePlataforma" type="button"
value="Subir Plataforma"
                    onclick="return SobePlataforma_onclick()" /></td>
            </tr>
            <tr>
                <td class="style1"><input class="bot" id="DescePlataforma" type="button"
value="Descer Plataforma"
                    onclick="return DescePlataforma_onclick()" /></td>
            </tr>
            <tr>
                <td class="style1"><input class="bot" id="SobeAltura" type="button"
value="Subir Medidor de Altura"
                    onclick="return SobeAltura_onclick()" /></td>
            </tr>
            <tr>
                <td class="style1"><input class="bot" id="DesceAltura" type="button"
value="Descer Medidor de Altura"
                    onclick="return DesceAltura_onclick()" /></td>
            </tr>
            <tr>
                <td class="style1"><input class="bot" id="Resetar" type="button" value="Posição
inicial"
                    onclick="return Resetar_onclick()" /></td>
            </tr>
            <tr>
                <td class="style1"><input class="bot" id="Automatico" type="button"
value="Automatico"
                    onclick="return Automatico_onclick()" /></td>
            </tr>

        </table>
    </div>

    <div class="meio">
        <div class="inputs">
            <div align="left" style="margin-left:10px;">INPUT</div>
            <div style="margin-left:35px;" >
                <table>
                    <tr>
                        <td style="width:150px;">
                            Cores Aceitas:</td>
                        <td style="width:60px;">
                            Preto <input id="corpreto" type="checkbox" onclick="mudaPreto();"
/></td>
```

```
                        <td style="width:60px;">
                            Rosa <input id="corrosa" type="checkbox" onclick="mudaRosa();"/></td>
                        <td style="width:60px;">
                            Prata <input id="corprata" type="checkbox"
onclick="mudaPrata();"/></td>
                    </tr>
                </table>
            </div>
        </div>
        <div class="style5">
            <script type="text/javascript">
            var height_array = new Array();
            var width_array = new Array();
            width_array[1] = 320;
            height_array[1] = 240;
        </script>
        <img src="http://192.168.0.193:8080/loading.jpg" class="webcam" id="webcam1"
onmousedown="PTZMouseDown1(event)" width="379" height="290" alt="Live Stream" />
        <script type="text/javascript">
            <!--
            currentCamera1 = 1;
            errorimg1 = 0;
            document.images.webcam1.onload = DoIt1;
            document.images.webcam1.onerror = ErrorImage1;
            function LoadImage1() {
                uniq1 = Math.random();
                document.images.webcam1.src = "http://192.168.0.193:8080/cam_" +
currentCamera1 + ".jpg?uniq=" + uniq1;
                document.images.webcam1.onload = DoIt1;
            }
            function PTZMouseDown1(e) {
                var IE = document.all ? true : false;
                var x, y;
                var myx, myy;
                var myifr = document.getElementById("_iframe-ptz");
                tp = getElPos1();
                myx = tp[0];
                myy = tp[1];
                if (IE) {
                    var scrollX = document.documentElement.scrollLeft ?
document.documentElement.scrollLeft : document.body.scrollLeft;
                    var scrollY = document.documentElement.scrollTop ?
document.documentElement.scrollTop : document.body.scrollTop;
                    x = event.clientX - myx + scrollX;
                    y = event.clientY - myy + scrollY;
                } else {
                    x = e.pageX - myx;
                    y = e.pageY - myy;
                }
                if ((width_array[currentCamera1] != null) && (width_array[currentCamera1] >
0)) x = Math.round((x * 400) / width_array[currentCamera1]);
                if ((height_array[currentCamera1] != null) && (height_array[currentCamera1] >
0)) y = Math.round((y * 300) / height_array[currentCamera1]);
                if (x > 400) x = 400;
                if (y > 300) y = 300;
                if (myifr != null) myifr.src = "http://192.168.0.193:8080/ptz?src=" +
currentCamera1 + "&moveto_x=" + x + "&moveto_y=" + y + "";
                return true;
            }
```

```
                function getElPos1() {
                    el = document.images.webcam1;
                    x = el.offsetLeft;
                    y = el.offsetTop;
                    elp = el.offsetParent;
                    while (elp != null) {
                        x += elp.offsetLeft;
                        y += elp.offsetTop;
                        elp = elp.offsetParent;
                    }
                    return new Array(x, y);
                }
                function ErrorImage1() {
                    errorimg1++;
                    if (errorimg1 > 3) {
                        document.images.webcam1.onload = "";
                        document.images.webcam1.onerror = "";
                        document.images.webcam1.src = "offline.jpg";
                    } else {
                        uniq1 = Math.random();
                        document.images.webcam1.src = "http://192.168.0.193:8080/cam_" +
currentCamera1 + ".jpg?uniq=" + uniq1;
                    }
                }
                function DoIt1() {
                    errorimg1 = 0;
                    window.setTimeout("LoadImage1();", 40);
                }
            //-->
            </script>
        </div>
    </div>

    <div class="direita">
        <div align="left" style="margin-left:23px;">OUTPUT</div>
        <div align="right" class="style2" style="margin-top:30px">
            <div class="style3">Controle WEB Habilitado</div>
            <div id="Controle_web" class="style6"></div>
        </div>
        <div align="right" class="style2" style="margin-top:15px">
            <div class="style3">Prata</div>
            <div id="Prata" class="style4"></div>
        </div>
        <div align="right" class="style2">
            <div class="style3">Rosa</div>
            <div id="Rosa" class="style4"></div>
        </div>
        <div align="right" class="style2">
            <div class="style3">Preto</div>
            <div id="Preto" class="style4"></div>
        </div>
        <div align="right" class="style2" style="margin-top:15px">
            <div class="style3">Plataforma Abaixada</div>
            <div id="Plat_baixo" class="style4"></div>
        </div>
        <div align="right" class="style2">
            <div class="style3">Plataforma Elevada</div>
            <div id="Plat_cima" class="style4"></div>
        </div>
```

```
            <div align="right" class="style2">
               <div class="style3">Cilindro Horizontal Recuado</div>
               <div id="Cil_hor" class="style4"></div>
            </div>
            <div align="right" class="style2">
               <div class="style3">Cilindro Vertical Abaixado</div>
               <div id="Cil_ver" class="style4"></div>
            </div>

         </div>
      </div>

   </div>
   </form>
</div>
</body>
</html>
```

codigoJS.js

```
var cpreto,crosa,cprata,conex;

function CarregaXML_onclick() {
   var webserv = new Service.WebServ();
   webserv.CarregaXml();
}
function ConectaOPC_onclick() {
   webserv = new Service.WebServ();
   webserv.ConnOpc();
   conex="true";
   cpreto = "false";
   crosa = "false";
   cprata = "false";
}
function ExpulsaPeca_onclick() {
   webserv = new Service.WebServ();
   webserv.avancaPeca();
}
function RecuaPeca_onclick() {
   webserv = new Service.WebServ();
   webserv.recuaPeca();
   cor = "true";
}
function SobePlataforma_onclick() {
   webserv = new Service.WebServ();
   webserv.sobePlataforma();
}
function DescePlataforma_onclick() {
   webserv = new Service.WebServ();
   webserv.descePlataforma();
}
function SobeAltura_onclick() {
   webserv = new Service.WebServ();
   webserv.sobeAltura();
}
function DesceAltura_onclick() {
   webserv = new Service.WebServ();
   webserv.desceAltura();
}
```

```
function Resetar_onclick() {
    webserv = new Service.WebServ();
    webserv.setStart();
}
function Automatico_onclick() {
    webserv = new Service.WebServ();
    webserv.automatico(cpreto, crosa, cprata, 30);
}

function atualizacao() {
    webserv = new Service.WebServ();
    if(conex=="true"){
        webserv.leituraComunWEB("R008", atualizaWEB);
        webserv.corPeca(atualizaCor);
        webserv.leituraSensor("R003", atualizaPlatBaixo);
        webserv.leituraSensor("R004", atualizaPlatCima);
        webserv.leituraSensor("R005", atualizaCilHor);
        webserv.leituraSensor("R006", atualizaCilVer);
        }
}

function atualizaCilVer(result) {
    if (result == "-1") {
        document.getElementById("Cil_ver").style.backgroundColor = "#00FF00";
    }
    else if (result == "0") {
    document.getElementById("Cil_ver").style.backgroundColor = "#FF0000";
    }
}
function atualizaCilHor(result) {
    if (result == "-1") {
        document.getElementById("Cil_hor").style.backgroundColor = "#00FF00";
    }
    else if (result == "0") {
        document.getElementById("Cil_hor").style.backgroundColor = "#FF0000";
    }
}
function atualizaPlatCima(result) {
    if (result == "-1") {
        document.getElementById("Plat_cima").style.backgroundColor = "#00FF00";
    }
    else if (result == "0") {
        document.getElementById("Plat_cima").style.backgroundColor = "#FF0000";
    }
}
function atualizaPlatBaixo(result) {
    if (result == "-1") {
        document.getElementById("Plat_baixo").style.backgroundColor = "#00FF00";
    }
    else if (result == "0") {
        document.getElementById("Plat_baixo").style.backgroundColor = "#FF0000";
    }

}

function mudaPreto() {
    if (cpreto == "false") {
        cpreto = "true";
    }
```

```
    else {
        cpreto = "false";
    }
}
function mudaRosa() {
    if (crosa == "false") {
        crosa = "true";
    }
    else {
        crosa = "false";
    }
}
function mudaPrata() {
    if (cprata == "false") {
        cprata = "true";
    }
    else {
        cprata = "false";
    }
}

function atualizaCor(result) {
    if(result!="mantercor"){
        if (result == "Preto") {
            document.getElementById("Preto").style.backgroundColor = "#00FF00";
            document.getElementById("Rosa").style.backgroundColor = "#FF0000";
            document.getElementById("Prata").style.backgroundColor = "#FF0000";

        }
        if (result == "Prata") {
            document.getElementById("Prata").style.backgroundColor = "#00FF00";
            document.getElementById("Preto").style.backgroundColor = "#FF0000";
            document.getElementById("Rosa").style.backgroundColor = "#FF0000";
        }
        if (result == "Rosa") {
            document.getElementById("Rosa").style.backgroundColor = "#00FF00";
            document.getElementById("Prata").style.backgroundColor = "#FF0000";
            document.getElementById("Preto").style.backgroundColor = "#FF0000";
        }
        if (result == "npeca") {
            document.getElementById("Preto").style.backgroundColor = "#FF0000";
            document.getElementById("Prata").style.backgroundColor = "#FF0000";
            document.getElementById("Rosa").style.backgroundColor = "#FF0000";
        }
    }
}

function atualizaWEB(result) {
    if (result == "-1") {
        document.getElementById("Controle_web").style.backgroundColor = "#00FF00";
    }
    else if (result == "0") {
        document.getElementById("Controle_web").style.backgroundColor = "#FF0000";
    }
    else {
        document.getElementById("Controle_web").style.backgroundColor = "#FFFF00";
    }
}
```

Apêndice D – Instalação elétrica