DESENVOLVIMENTO DE UM SUPERVISÓRIO MODULAR PARA UMA CÉLULA FLEXÍVEL DE MANUFATURA

UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM
ENGENHARIA MECÂNICA

DESENVOLVIMENTO DE UM SUPERVISÓRIO MODULAR PARA UMA CÉLULA FLEXÍVEL DE MANUFATURA

Dissertação submetida à

UNIVERSIDADE FEDERAL DE SANTA CATARINA

para a obtenção do grau de

MESTRE EM ENGENHARIA MECÂNICA

HUGO GASPAR SANTOS

Florianópolis, Fevereiro de 2007

UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM
ENGENHARIA MECÂNICA

DESENVOLVIMENTO DE UM SUPERVISÓRIO MODULAR
PARA UMA CÉLULA FLEXÍVEL DE MANUFATURA

HUGO GASPAR SANTOS

Esta dissertação foi julgada adequada para a obtenção do título de

MESTRE EM ENGENHARIA
ESPECIALIDADE ENGENHARIA MECÂNICA

Sendo aprovada em sua forma final.

Prof. João Carlos Espíndola Ferreira, Ph.D. Orientador - UFSC	Prof. Marcelo Teixeira dos Santos, Dr. Co-Orientador - IST

Prof. Fernando Cabral, Ph.D.
Coordenador do Curso

BANCA EXAMINADORA

Prof. Jonny Carlos da Silva, Dr.
UFSC

Prof. Ricardo José Rabelo, Dr. UFSC	Prof. Victor Juliano De Negri, Dr. UFSC

AGRADECIMENTOS

Aos professores orientadores
João Carlos Espíndola Ferreira
e Marcelo Teixeira dos Santos,
pela confiança e apoio.
Aos professores
Léo Schirmer, André Garcia,
Stéfano, Wanilson,
Alberto, Maikon e Ruthes,
pelo valioso auxílio no projeto.
Aos colegas
Alexandre Fleig, Vadis Bellini,
Ewandro, Kelly, William, Sérgio e Leônidas,
pelas incontáveis caronas a Florianópolis.
Aos amigos
Laudelino, Mikos, Reaes, Mário Mello,
Izabel Zattar, Júlio Ticona e Luis Pepplow,
pela inestimável e rica troca de experiências.
À SOCIESC,
na pessoa do professor Cláudio von Dokonal
pela permissão do uso do laboratório.
À Elipse,
pelo suporte e gentil empréstimo do hardkey.
À CAPES,
pelo imprescindível suporte financeiro.
À UFSC,
por conceder a oportunidade
da realização deste trabalho.

DEDICATÓRIA

À minha esposa Patrícia
por todo o amor e
pelo inestimável apoio ao longo da caminhada.
Aos meus filhos Anna, Álex e Dimitri,
pelo carinho com que me brindaram
durante todo esse tempo.

RESUMO

Este trabalho tem o objetivo de propor e implementar um conjunto de instruções e procedimentos para integrar, de forma física e lógica, um grupo de máquinas CNC, um robô industrial e um sistema de armazenamento automático (*Automated Storage and Retrieval System* - AS/RS), para a formação de uma Célula Flexível de Manufatura (*Flexible Manufacturing Cell* - FMC). Em primeiro lugar foi feita a modelagem da célula, com base na definição prévia de um conjunto de interfaces mínimas para cada equipamento e com o auxílio do mecanismo formal de descrição das Redes de Petri. A partir da modelagem da célula, foi proposto e implementado um modelo de integração, baseado em módulos gerenciadores desenvolvidos a partir de um *software* de supervisão e controle (Elipse SCADA). Cada módulo foi concebido com interfaces suficientes para permitir a integração de equipamentos de diferentes fabricantes, o que confere flexibilidade na configuração da célula. A comunicação entre os diversos módulos é realizada utilizando a tecnologia OPC (OLE *for Process Control*), que possibilita a troca de dados entre gerenciadores em tempo real e de forma aberta. A integração entre os gerenciadores e seus respectivos equipamentos é feita por meio de CLPs (Controladores Lógicos Programáveis). O sistema permite simular a fabricação de uma família de peças, sem a interferência de operadores. Interfaces Homem-Máquina (IHM) foram desenvolvidas para permitir que usuários locais e remotos (fisicamente distantes) possam inserir e monitorar seus pedidos de fabricação na célula. Todos os códigos utilizados na construção dos gerenciadores são descritos neste trabalho, além dos programas de movimentação do robô, programas desenvolvidos para os CLPs e implementações de *hardware* necessários para a integração dos diferentes equipamentos que compõem a célula.

Palavras-chaves: Célula Flexível de Manufatura; Sistema Flexível de Manufatura; Modelagem de sistemas de manufatura; Sistemas de supervisão e controle.

ABSTRACT

This work aims at proposing and implementing a set of instructions and procedures to integrate, physical and logically, a group of CNC (Computer Numerical Control) machines, an industrial robot and an Automatic Storage and Retrieval System (AS/RS), in order to form a Flexible Manufacturing Cell (FMC). First of all, the cell was modeled based on a definition of a set of minimum interfaces for each piece of equipment, using Petri Nets. After completing the modeling of the cell, it was proposed and implemented an integration model, based on management modules developed with a monitoring and control software (Elipse SCADA - Supervisory Control and Data Acquisition). Each module was conceived with enough interfaces to allow the integration of different equipment manufacturers, which ensures flexibility in the cell configuration. The communication between the several modules is made using OPC (OLE for Process Control) technology, which allows the data exchange between the management modules real time and in an open way. The integration between managers and their equipments is done by a PLC (Programmable Logic Controller). The system admits to simulate a piece family manufacturing without any operator interference. HMIs (Human Machine Interfaces) were developed to allow local and remote (physically distant) users to insert their manufacture order and to monitor them. All codes used in the development of the management modules are described in this work, and so the robot movement programs, the programs developed for the CLPs, and the hardware implementations necessary for the integration of the different pieces of equipment that compose the cell.

Keywords: Flexible manufacturing cell; Flexible manufacturing system; Manufacturing system modeling; Supervisory control and data acquisition.

RESUMEN

Este trabajo tiene el objetivo de proponer e implementar un conjunto de instrucciones y procedimientos para integrar, de forma física y lógica, un grupo de máquinas CNC, un robot industrial y un sistema de almacenamiento automático (*Automated Storage and Retrieval System* – AS/RS), para la formación de una Celda Flexible de Manufactura (*Flexible Manufacturing Cell* – FMC). En primer lugar se efectuó el modelado de la celda, con base en la definición previa de un conjunto de interfaces mínimas para cada equipo y con el auxilio del mecanismo formal de descripción de las Redes de Petri. A partir del modelado de la celda, se propone y se implementa un modelo de integración, basado en módulos de gerencia desarrollados a partir de un *software* de supervisión y control (Elipse SCADA). Cada módulo es concebido con interfaces suficientes para permitir la integración de equipos de diferentes fabricantes, lo que concede flexibilidad en la configuración de la celda. La comunicación entre los diversos módulos se realiza utilizando la tecnología OPC (OLE *for Process Control*), que permite el intercambio de datos entre módulos de gerencia en tiempo real y de forma abierta. La integración entre los módulos de gerencia y sus respectivos equipos se implementa por medio de CLPs (Controladores Lógicos Programables). El sistema permite simular la fabricación de una familia de piezas, sin interferencia de operadores. Interfaces Hombre-Máquina (IHM) son desarrolladas para permitir que usuarios locales y remotos (físicamente distantes) puedan incluir y monitorear sus pedidos de fabricación en la celda. Todos los códigos utilizados en la construcción de los módulos de gerencia son detallados en este trabajo, además de los programas de movimiento del robot, programas desarrollados para los CLPs e implantaciones de *hardware* necesarios para la integración de los diferentes equipos que componen la celda.

Palabras Llaves: Celda Flexible de Manufactura; Sistema Flexible de Manufactura; Modelado de sistemas de manufactura; Sistemas de supervisión y control.

SUMÁRIO

ÍNDICE DE FIGURAS xiii
ÍNDICE DE TABELAS xvi
LISTA DE SÍMBOLOS, NOMENCLATURAS E ABREVIAÇÕES xvii

1. INTRODUÇÃO 1
1.1 PROBLEMA DE PESQUISA 2
1.2 OBJETIVO GERAL DO TRABALHO 3
1.3 OBJETIVOS ESPECÍFICOS 4
1.4 JUSTIFICATIVA 5
1.5 METODOLOGIA 7
1.5.1 Classificação da Pesquisa 7
1.5.2 Pressupostos básicos 8
1.5.3 Roteiro da pesquisa 9
1.6 DELIMITAÇÕES DO TRABALHO 10
1.7 ESTRUTURA DA DISSERTAÇÃO 11

2. REVISÃO BIBLIOGRÁFICA: HISTÓRICO, SISTEMAS, OPC E SUPERVISÓRIOS 13
2.1 HISTÓRICOS DOS SISTEMAS DE FABRICAÇÃO 13
2.2 ARQUITETURA PARA SISTEMAS DE MANUFATURA 15
2.2.1 Sistema Funcional 16
2.2.2 Sistema em Linha 17
2.2.3 Sistema Celular 18
2.2.4 Sistemas Flexíveis de Manufatura 18
2.2.5 Manufatura Integrada por Computador 21
2.2.6 Comparativo entre os sistemas de manufatura 24
2.3 SISTEMAS DE SUPERVISÃO E CONTROLE 25
2.3.1 Componentes Físicos de um Sistema SCADA 29
2.3.2 Componentes Lógicos de um Sistema SCADA 30
2.4 PADRÃO “OLE for PROCESS CONTROL” (OPC) 31

2.5 SISTEMAS SCADA COM OPC 33
2.6 *SOFTWARE* ELIPSE SCADA 35
2.6.1 Aplicativos no Elipse SCADA 36
2.6.2 OPC no Elipse SCADA 38

3. DESCRIÇÃO GERAL DO TRABALHO 41
3.1 O LABORATÓRIO 41
3.2 DISTRIBUIÇÃO DOS EQUIPAMENTOS NO LABORATÓRIO 42
3.2.1 Armazém Automático 42
3.2.2 Robô ABB 43
3.2.3 Equipamentos CNC 44
3.3 O DESAFIO DA INTEGRAÇÃO 45
3.4 ROTEIRO DE TRABALHO 48

4. MODELAGEM DA FMC 53
4.1 A CÉLULA FLEXÍVEL DE MANUFATURA 53
4.2 INTERFACES 54
4.2.1 Interfaces dos Equipamentos CNC 56
4.2.2 Interfaces do Robô 57
4.2.3 Interfaces do Armazém Automático 58
4.2.4 Interfaces do Gerenciador FMC 59
4.3 MODELAGEM A EVENTOS DISCRETOS - REDES DE PETRI 60
4.4 CONVERSÃO DAS RdPI PARA CÓDIGOS DE CONTROLE SUPERVISÓRIO 66

5. INTEGRAÇÃO DA CÉLULA 68
5.1 ARQUITETURA DA CÉLULA 68
5.2 GERENCIADOR DO TORNO 69
5.2.1 Aplicativo gerenciador do torno no Elipse SCADA 71
5.2.2 Integração do gerenciador do torno com CLP Moeller 76
5.2.3 Integração entre CLP Moeller e o torno 80
5.2.4 Integração entre gerenciador, CLP Moeller e torno 84
5.3 GERENCIADOR DA FRESADORA E GERENCIADOR DO CNC R1 85
5.4 GERENCIADOR DO ROBÔ 89

5.4.1 Aplicativo gerenciador do robô no Elipse SCADA 91
5.4.2 Integração do gerenciador do robô com CLP Siemens 95
5.4.3 Integração entre CLP Siemens e o robô 98
5.4.4 Integração do gerenciador, CLP Siemens e robô 104
5.5 GERENCIADOR DO ARMAZÉM E GERENCIADOR FMC 106
5.5.1 Aplicativo gerenciador do armazém e da FMC no Elipse SCADA 108
5.6 GERENCIADOR REMOTO 114
5.6.1 Gerenciador remoto no Elipse SCADA 115
5.7 INTEGRAÇÃO GERAL DA CÉLULA 117
5.8 FUNCIONAMENTO DA CÉLULA 119

6. CONCLUSÕES, CONTRIBUIÇÕES E TRABALHOS FUTUROS 122
6.1 CONCLUSÕES 122
6.2 CONTRIBUIÇÕES 124
6.3 TRABALHOS FUTUROS 125

REFERÊNCIAS BIBLIOGRÁFICAS 127

APÊNDICE 135

APÊNDICE A – RdPI DA CÉLULA FLEXÍVEL DE MANUFATURA 135
A1. Redes de Petri Interpretadas da integração da mesa1 do AS/RS com o torno 135
A2. Redes de Petri Interpretadas da integração da mesa2 do AS/RS com o torno 136
A3. Redes de Petri Interpretadas da integração da mesa1 do AS/RS com a fresadora 136
A4. Redes de Petri Interpretadas da integração da mesa2 do AS/RS com a fresadora 137
A5. Redes de Petri Interpretadas da integração da mesa1 do AS/RS com o CNC R1 137
A6. Redes de Petri Interpretadas da integração da mesa2 do AS/RS com o CNC R1 138
A7. Redes de Petri Interpretadas da integração das mesas do AS/RS com o torno 138
A8. Redes de Petri Interpretadas da integração do torno com a mesa1 do AS/RS 139
A9. Redes de Petri Interpretadas da integração do torno com a mesa2 do AS/RS 139
A10. Redes de Petri Interpretadas da integração das mesas do AS/RS com a fresadora 140
A11. Redes de Petri Interpretadas da integração da fresadora com a mesa1 do AS/RS 141
A12. Redes de Petri Interpretadas da integração da fresadora com a mesa2 do AS/RS 141

A13. Redes de Petri Interpretadas da integração das mesas do AS/RS com o CNC R1 142
A14. Redes de Petri Interpretadas da integração do CNC R1 com a mesa1 do AS/RS 143
A15. Redes de Petri Interpretadas da integração do CNC R1 com a mesa2 do AS/RS 143

APÊNDICE B – DIAGRAMAS ELÉTRICOS MODIFICADOS NO TORNO 144

APÊNDICE C – PROGRAMAÇÃO DO ROBÔ 147
C1. Tabela da seqüência de pontos dos programas do robô 147
C2. Posições relevantes da movimentação do robô 147
C3. Fluxograma dos programas de movimentação do robô 148
C4. Programas do robô 149

APÊNDICE D – *SCRIPTS* DE PROGRAMAÇÃO 155
D1. *SCRIPT* DE PROGRAMAÇÃO DO ROBÔ 155
D2. *SCRIPT* DE PROGRAMAÇÃO DA FRESADORA 157
D3. *SCRIPT* DE PROGRAMAÇÃO DO GERENCIADOR REMOTO 160
D2. *SCRIPT* DE PROGRAMAÇÃO DO GERENCIADOR DO ARMAZÉM E DA FMC ... 161

ÍNDICE DE FIGURAS

Figura 2.1. *Layout* típico do sistema funcional 16
Figura 2.2. *Layout* típico de um sistema em linha..... 17
Figura 2.3. *Layout* de um sistema celular..... 18
Figura 2.4. Exemplo de um FMS..... 19
Figura 2.5. Funcionamento de um sistema CIM..... 22
Figura 2.6. Diagrama entidade/relacionamento da tecnologia do CIM..... 23
Figura 2.7. Características de capacidade produtiva e flexibilidade dos sistemas de produção..... 25
Figura 2.8. Sistema de Supervisão e Controle..... 30
Figura 2.9. Camadas de informação englobadas pela tecnologia OPC 32
Figura 2.10. Arquitetura de um sistema de gerenciamento de processos industriais 33
Figura 2.11. Arquitetura entre Cliente OPC e diversos Servidores OPC 34
Figura 2.12. Exemplo de relacionamento entre Servidores e Clientes OPC 35
Figura 2.13. Barra de tarefas e propriedades do objeto botão 37
Figura 2.14. *Organizer* e tipos de *tags* do Elipse SCADA 37
Figura 2.15. *Drivers* e configuração no *organizer* do Elipse SCADA..... 38
Figura 2.16. Lista de Servidores OPC e configuração..... 39
Figura 2.17. Importação de uma *tag* do servidor OPC..... 39
Figura 2.18. Configuração do DCOM 40
Figura 3.1. Vista parcial do laboratório..... 41
Figura 3.2. Armazém automático 43
Figura 3.3. Controlador e unidade de programação do robô 44
Figura 3.4. Torno e centro de usinagem CNC..... 44
Figura 3.5. Arquitetura da proposta de integração da célula 47
Figura 3.6. Evolução histórica das arquiteturas dos sistemas de controle 48
Figura 3.7. Arquitetura para integração da FMC 51
Figura 4.1. Identificação dos equipamentos da célula..... 54
Figura 4.2. Conexão entre gerenciador, CLP e equipamento da FMC..... 55
Figura 4.3. Interfaces dos equipamentos CNC 56

Figura 4.4. Interfaces do robô 58
Figura 4.5. Interfaces do armazém automático 59
Figura 4.6. Interfaces do gerenciador FMC 60
Figura 4.7. RdPI simplificada para movimentação de peças da mesa1 até o torno 64
Figura 4.8. Exemplo de conversão da RdPI para o código de controle 67
Figura 5.1. Arquitetura da FMC 69
Figura 5.2. *Tags* utilizadas pelo gerenciador do torno 71
Figura 5.3. Configuração das *tags* OPC no *organizer* do Elipse SCADA 72
Figura 5.4. *Tags* do gerenciador do torno 74
Figura 5.5. Interface com operador do gerenciador do torno 76
Figura 5.6. *Driver* e configuração do CLP Klockner Moeller 77
Figura 5.7. Parâmetros "N" da configuração do CLP Klockner Moeller 78
Figura 5.8. Programa do CLP Moeller do torno 79
Figura 5.9. Esquema de conexões do CLP Moeller 80
Figura 5.10. Cilindro pneumático, porta do torno e banco de relés auxiliares (no detalhe) 81
Figura 5.11. *Ladder* fechar e abrir porta 81
Figura 5.12. *Ladder* do torno livre/usinando 82
Figura 5.13. *Ladder* da castanha aberta/fechada 83
Figura 5.14. *Ladder* de robô em área segura e *start* 83
Figura 5.15. Arquitetura da troca de informações entre gerenciador, CLP Moeller e torno 85
Figura 5.16. CLP Moeller, banco auxiliar de relés e conexões 85
Figura 5.17. *Tags* utilizadas pelos gerenciadores da fresadora e do CNC R1 87
Figura 5.18. Interface com o operador do gerenciador da fresadora e *tags* do *organizer* 88
Figura 5.19. Conjunto de *tags* utilizado pelo gerenciador do robô 90
Figura 5.20. Interface do gerenciador do robô com operador 92
Figura 5.21. Estrutura das *tags* OPC e detalhe de um *script* 93
Figura 5.22. *Tags* utilizadas pelo gerenciador do robô e parametrização do *driver* Prodave 95
Figura 5.23. Programa do CLP Siemens e topologia da rede 96
Figura 5.24. Conexões entre gerenciador, CLP Siemens e robô 97
Figura 5.25. Arquitetura do protocolo Profibus 99
Figura 5.26. Posições da trajetória do robô, CLP Siemens e controlador do robô 104
Figura 5.27. *Script* retorno da peça da usinagem 105

Figura 5.28. Berços de usinagem, saída/rejeito e entrada 107
Figura 5.29. Interfaces para os gerenciadores FMC e do armazém 107
Figura 5.30. Cadastro de peças e processos no gerenciador do armazém 109
Figura 5.31. Tela de solicitação de usinagem 110
Figura 5.32. *Script* associado à *tag* presença_mesa1 111
Figura 5.33. *Script* para atualização da *tag* ProgUsinagem após o início da usinagem 112
Figura 5.34. Estruturas dos servidores OPC e uma tela do gerenciador FMC 113
Figura 5.35. Estrutura desmembrada das *tags* OPC e tela principal do aplicativo 114
Figura 5.36. Interfaces do gerenciador remoto 115
Figura 5.37. Estrutura das *tags* do gerenciador remoto e interface com usuário 116
Figura 5.38. Cadastro de usuários e autenticação da aplicação 117
Figura 5.39. Arquitetura das conexões na célula e recursos de *software* 118

ÍNDICE DE TABELAS

Tabela 2.1. Comparativo entre sistemas para diferentes volumes de produção 24
Tabela 3.1. Relação entre peças, códigos, mesas e CNC 51
Tabela 5.1. Conversão de interfaces do torno para *tags* 70
Tabela 5.2. Significado das *tags* para o gerenciador do torno 70
Tabela 5.3. *Script* da *tag* ProgUsinagem 73
Tabela 5.4. *Script* da *tag* SINAIS_DO_ROBO – OnValueChanged 73
Tabela 5.5. Relação entre *tags* e mensagens ao usuário 74
Tabela 5.6. *Script* da *tag* Torno_Moeller 75
Tabela 5.7. Configuração dos Parâmetros "N" 78
Tabela 5.8. Relação entre *tags*, sinais do CLP Moeller e programas 83
Tabela 5.9. Conversão de interfaces da fresadora e do R1 para *tags* 86
Tabela 5.10. Significado das *tags* para os gerenciadores da fresadora e R1 86
Tabela 5.11. Relação entre *tags* e ações correspondentes 88
Tabela 5.12. Conversão de interfaces do robô para *tags* 89
Tabela 5.13. Relação entre *tags*, mesa de usinagem, CNC e programa do robô 94
Tabela 5.14. Configuração dos Parâmetros "N" para o *driver* Prodave 96
Tabela 5.15. Arquitetura do protocolo Profibus 98
Tabela 5.16. Relação entre *tags* PLC e I/Os digitais do robô 99
Tabela 5.17. Programa principal do robô 100
Tabela 5.18. Programas de carga e descarga do torno a partir da mesa1 101
Tabela 5.19. Relação entre a *tag* AuxiliarProg, mesa de usinagem e programa do robô 106
Tabela 5.20. Conversão de interfaces do armazém e FMC para *tags* 108
Tabela 5.21. Relação de código de peças 110
Tabela 5.22. Conversão de interfaces do gerenciador remoto para *tags* 115

LISTA DE SÍMBOLOS, NOMENCLATURAS E ABREVIAÇÕES

API	- *Application Programming Interface*
AS/RS	- Armazém automático (*Automatic Storage/ Retrieval System*)
BCD	- *Binary Coded Decimal*
BD	- Banco de Dados
CAD	- *Computer Aided Design*
CAM	- *Computer Aided Manufacturing*
CAPP	- *Computer Aided Process Plan.*
CIM	- *Computer Integrated Manufacturing*
CLP	- Controlador Lógico Programável
CNC	- Comando Numérico Computadorizado
COM	- *Component Object Model*
DCOM	- *Distributed Component Object Model*
DDLM	- *Direct Data Link Mapper*
ERP	- *Enterprise Resource Planning*
FMC	- *Flexible Manufacturing Cell*
FMS	- *Flexible Manufacturing System*
FMSp	- *Fieldbus Message Specification*
HMI	- *Human-Machine Interface*
MES	- *Manufacturing Execution Systems*
MTU	- *Master Terminal Unit*
OLE	- *Object Linking and Embedding*
OPC	- *OLE for Process Control*
RAP	- *Robot Application Protocol*
RdP	- Redes de Petri
RdPI	- Redes de Petri Interpretadas
RTU	- *Remote Terminal Units*
SCADA	- *Supervisory Control and Data Acquisition*
SED	- Sistemas a Eventos Discretos
TCP/IP	- *Transmission Control Protocol/Internet Protocol*

1. INTRODUÇÃO

A manufatura, termo que originalmente significa “fazer a mão” vem sendo afetada por profundas mudanças nas últimas décadas. A competição globalizada, um fenômeno recente, somada à crescente demanda da sociedade por novidades, produtos personalizados, qualidade elevada e preços menores, obriga as empresas a modernizarem seus métodos produtivos. Estas mudanças não se processam mais como um diferencial de uma empresa em relação à outra, mas sim como fator de sobrevivência num mercado acirrado e competitivo. O diferencial tecnológico que algumas corporações detinham no passado, e que lhes permitia produzir determinados produtos com características únicas, é cada vez mais raro nos dias atuais. No passado o consumidor se adaptava aos produtos oferecidos pelas empresas. Atualmente são as empresas que necessitam entender e adaptar-se às exigências dos consumidores.

Sistemas produtivos que priorizam o volume de produção, com custo baixo e pequena variedade, estão enfrentando o desafio de flexibilizar sua produção e diversificar seus produtos, sem originar um acréscimo significativo nos custos. Para pequenos e medianos lotes de produção, os Sistemas Flexíveis de Manufatura (FMS) são uma alternativa viável de resposta aos desafios de flexibilidade, diversidade e custo acessível.

O conceito de FMS envolve um alto nível de automação, onde a fabricação é realizada por centros de usinagem automatizados e a transferência entre as máquinas é feita por veículos auto-guiados, ou esteiras automatizadas. O posicionamento e movimentação das peças são realizados por robôs manipuladores. Os produtos acabados e a matéria-prima são estocados em um armazém automático. Os recursos produtivos estão ligados a um sistema computacional que coordena todas as ações, capacitando o sistema a produzir diferentes tipos de peças, utilizando os mesmos equipamentos e o mesmo sistema de controle.

A complexidade técnica e os altos custos de aquisição e implantação dos FMSs dificultam sobremaneira o uso destes sistemas em empresas e instituições de ensino e pesquisa no país. O objetivo deste trabalho é oferecer uma contribuição ao desenvolvimento dessa nova tecnologia, através da integração física e lógica dos equipamentos existentes no laboratório da SOCIESC (Sociedade Educacional de Santa Catarina), para a construção de uma Célula Flexível de Manufatura (FMC).

1.1 PROBLEMA DE PESQUISA

A integração e flexibilização da manufatura é um dos principais problemas que a indústria que se moderniza é obrigada a enfrentar na atualidade. Ferreira (1998), ao abordar algumas das características que limitam o uso de Sistemas Flexíveis de Manufatura, afirma que tais sistemas são caros para projetar, além de complexos de analisar e controlar. A maior parcela da dificuldade do desenvolvimento de sistemas de controle na indústria de manufatura flexível pode ser atribuída aos altos custos envolvidos no desenvolvimento e manutenção do *software* de controle e à dificuldade de se conseguir definir a forma de integração dos sistemas de chão-de-fábrica, afirma Friedrich (1996). Ainda segundo o autor, os equipamentos de produção, como máquinas ferramentas, robôs e armazéns automáticos, assim como computadores e redes de comunicação, são encontrados com facilidade, contudo o *software* de controle e o modelo de integração para estes equipamentos não está prontamente disponível. Estes sistemas são multidisciplinares, envolvendo conhecimentos de manufatura, programação de computadores, análise e especificações de sistemas e redes de comunicação. Para Lepikson (2005), as empresas que pretendem atuar na área de manufatura moderna, atendendo a um mercado consumidor cada vez mais exigente e ávido de novidades, precisam estabelecer novos princípios para seus sistemas produtivos, entre os quais: modularidade, integração, controle adaptativo e ambiente distribuído e heterogêneo.

Atualmente há duas soluções possíveis para quem deseja instalar um FMS. A primeira delas é adquirir uma solução pronta de empresas fornecedoras, como a Yamazaki Mazak (http://www.mazak.jp), Kearney & Trecker Company (http://www.equipmentmls.com), SMC Internacional Training (http://www.smceu.com), Heller (http://www.heller-machinetools.com), entre outras. Neste caso, geralmente não ocorre a transferência de tecnologia para o usuário, pois os sistemas são proprietários e fechados. A segunda opção é construir um FMS a partir da integração física e lógica de um conjunto de máquinas de processamento (CNC), manuseio e transporte (robôs e esteiras automatizadas) e armazenamento (armazém automático). Para este segundo caso é necessário definir um conjunto de interfaces para os equipamentos, modelar o sistema de manufatura, estabelecer um padrão de integração para os equipamentos heterogêneos, um padrão de comunicação, um sistema de controle e um sistema computacional que gerencie todo o sistema e cumpra a função de interface com o usuário. Esta dissertação propõe e implementa um modelo para construção de um FMS nos moldes abordados pelo segundo caso.

Nos últimos anos, muitos e meritórios trabalhos foram publicados a respeito de flexibilidade e integração da manufatura, sobre modelagem, simulação e controle de FMSs, e sobre as dimensões da flexibilidade (de produto, sequenciamento, volume, processo, produção, etc.). Em contrapartida, raras são as publicações que descrevem em detalhes como integrar efetivamente os diversos recursos produtivos para a formação de uma Célula, ou Sistema Flexível de Manufatura.

Diante do cenário apresentado, e considerando que a pesquisa é um conjunto de ações e propostas para encontrar a solução para um problema, e que a pesquisa aplicada tem como objetivo gerar conhecimentos para aplicação prática e dirigidos à solução de problemas específicos, segundo afirmam Silva e Menezes (2005), pode-se formular o problema de pesquisa desta dissertação da seguinte forma:

- Como integrar de forma física e lógica os diversos equipamentos existentes no laboratório da SOCIESC, utilizando uma filosofia modular, aberta e expansível para formar uma Célula Flexível de Manufatura que possa simular a fabricação de uma família de peças de forma autônoma?

1.2 OBJETIVO GERAL DO TRABALHO

O objetivo principal desta dissertação é propor e implementar um modelo para integrar de forma física e lógica os equipamentos existentes no laboratório da SOCIESC e construir uma Célula Flexível de Manufatura que esteja apta a fabricar uma família de peças de forma autônoma. O sistema deve controlar a movimentação de materiais entre o armazém automático, o robô e as máquinas CNC, o sincronismo das ações necessárias para a efetiva fabricação das peças, assim como o controle dos estados de cada equipamento. Usuários locais e remotos (isto é, aqueles que se encontram fisicamente distantes da célula) podem inserir seus pedidos de fabricação de peças no sistema.

A integração é realizada através da conexão física e lógica dos equipamentos que compõem a célula. São desenvolvidos módulos ditos gerenciadores, um para cada equipamento (armazém automático, robô e CNCs), um para a célula (FMC) e outro para o usuário remoto. Os módulos são construídos como aplicativos do *software* supervisório Elipse SCADA (*Supervisory Control and Data Aquisition* – http://www.elipse.com.br) e trocam dados entre si em tempo real através de uma rede *Ethernet*, utilizando o padrão de comunicação industrial OPC (OLE *for Process Control*). A

comunicação dos gerenciadores com os equipamentos de produção, movimentação e fornecimento de material, utilizam CLPs (Controladores Lógicos Programáveis).

1.3 OBJETIVOS ESPECÍFICOS

Para atingir o objetivo principal do projeto, foram definidos os seguintes objetivos específicos:

- Decompor as ações executadas por cada equipamento na célula e as interações coletivas, para obter o mínimo de interfaces necessárias para uma operação conjunta;
- Modelar a FMC, um Sistema a Eventos Discretos (SED), utilizando o mecanismo formal de descrição das Redes de Petri Interpretadas (RdPI);
- Construir módulos gerenciadores para cada equipamento existente na célula, além de um módulo gerenciador para toda a FMC, utilizando o *software* supervisório Elipse SCADA;
- Integrar os módulos gerenciadores e os equipamentos de processamento, manipulação e fornecimento por meio de CLPs;
- Projetar e modificar os CNCs para dotá-los de funcionalidades que permitam autonomia operacional (por exemplo: automatizar abertura e fechamento de portas e dispositivos de fixação);
- Sincronizar todos os eventos e ações da célula, criando intertravamentos que forneçam segurança operacional e evitem riscos de colisão entre as partes móveis dos equipamentos;
- Integrar os diversos gerenciadores por meio de uma rede *Ethernet*, utilizando o padrão de comunicação industrial OPC;
- Criar uma interface homem-máquina que permita ao usuário do sistema inserir pedidos de fabricação de peças no sistema. Este aplicativo também deve controlar a entrada de matéria-prima no armazém, a saída de peças já processadas, os eventuais alarmes, emitir relatório de peças produzidas e estocadas, o estado de cada recurso produtivo da célula, configurar a estratégia de movimentação do armazém e a codificação de materiais e processos de transformação;
- Permitir que usuários fisicamente distantes também possam inserir seus pedidos de fabricação na célula, através de um gerenciador remoto;

- Simular a fabricação de uma família de peças na Célula Flexível de Manufatura, compostas pelas peças que compõem o jogo de xadrez (peão, torre, cavalo, bispo, dama e rei);
- Gerar uma bibliografia que reúna revisão bibliográfica, informações de cada equipamento, programas do robô, programas dos CLPs, modelagem da célula, gerenciadores e *scripts* associados, para fornecer subsídios que permitam uma continuidade no projeto de pesquisa e integração da FMC.

1.4 JUSTIFICATIVA

Este trabalho pode ser justificado pelas óticas da relevância e da contribuição. Do ponto de vista da relevância, destaca-se o número de instituições de ensino e pesquisa que estão atualmente instalando laboratórios de manufatura flexível, tanto no país quanto no exterior. Também é grande o número de publicações na área da flexibilidade e integração da manufatura, que atestam a importância do assunto para a área acadêmica e para a indústria (ZHOU e VENKATESH, 2000), (COINBRA et al, 2004), (LEPIKSON, 2005), (PINA et al, 2006), (VIEIRA, 1996), (RIBEIRO e ELVAS, 2004), (BOARETTO et al, 2004), (TEIXEIRA, 2005 e 2006), (FERREIRA, 2003_a e 2003_b), (FONSECA, 2002) e (ALVES et al, 2003).

O laboratório onde é implementado o projeto de integração, apesar de contar com todos os equipamentos necessários para a formação de uma FMC, de fato não possuía nenhum tipo de integração que permitisse a operação conjunta e autônoma dos equipamentos. Com a integração da célula disponibiliza-se um laboratório de manufatura integrada para uso em diversas disciplinas de graduação e pós-graduação na SOCIESC.

Do ponto de vista da contribuição, esta dissertação propõe e implementa um modelo de integração, composto por um conjunto de instruções e procedimentos, para a construção de uma Célula Flexível de Manufatura. Os principais elementos utilizados para projetar e construir a FMC são: a modelagem da célula em Redes de Petri Interpretadas, a construção de módulos gerenciadores utilizando um sistema de supervisão e controle (SCADA) e a integração destes módulos, por meio do padrão de comunicação industrial OPC, em um sistema de controle heterárquico. As Redes de Petri são utilizadas para modelar sistemas a eventos discretos, como é o caso de FMCs e FMSs (MUŠIČ e MATKO, 1998), (ZHOU e VENKATESH, 2000), (VIEIRA, 1996), (TENG e ZHANG, 1993),

(RIBEIRO e ELVAS, 2004) e (TEIXEIRA, 2006). Os sistemas supervisórios são amplamente utilizados para controle de processos industriais (ALVES et al, 2003), (BARROS, 2005), (PIRES et al, 2005), (SALVADOR, 2005) e (ELIPSE, – http://www.elipse.com.br). O padrão de comunicação OPC ganha importância e é cada vez mais utilizado na indústria por permitir que aplicações de *software* troquem dados entre si de forma aberta e simplificada, assim como aplicações gerenciais e dados de chão-de-fábrica (OPC Foundation, 1998), (FONSECA, 2002), (GAIDZINSKI, 2003), (ALVES et al, 2003), (NASCIMENTO, 2005) e (SALVADOR, 2005). A utilização de um sistema de controle heterárquico (PELS et al, 1997), FRIEDRICH (1996) e (TEIXEIRA, 2006) permite um elevado grau de autonomia aos diferentes módulos gerenciadores, que atuam todos no mesmo nível hierárquico e em estreita cooperação.

Todos os elementos utilizados para efetuar a integração física e lógica dos equipamentos já são conhecidos no meio industrial e acadêmico, assim como o uso de CLPs para integrar módulos gerenciadores com equipamentos produtivos. A combinação destes elementos, entretanto, compõe um detalhado roteiro para formação de uma FMC modular, aberta, expansível, com ambiente distribuído e heterogêneo.

1.5 METODOLOGIA

A metodologia aplicada a esta dissertação está dividida em: classificação da pesquisa, pressupostos básicos e roteiro da pesquisa, tópicos que serão abordados a seguir.

1.5.1 Classificação da Pesquisa

Segundo Silva e Menezes (2000), existem várias formas de classificar a pesquisa. As formas clássicas são as seguintes:

- Quanto à natureza: do ponto de vista da natureza a pesquisa pode ser básica ou aplicada (SILVA e MENEZES, 2000). Esta dissertação é aplicada porque tem o objetivo de integrar de forma física e lógica os diferentes equipamentos do laboratório da SOCIESC para a formação de uma Célula Flexível de Manufatura.

- Quanto à forma de abordagem do problema: do ponto de vista da abordagem do problema a pesquisa pode ser quantitativa ou qualitativa (SILVA e MENEZES, 2000). Este trabalho de pesquisa é eminentemente qualitativo, pois utiliza o ambiente natural como fonte de dados. A interpretação dos fenômenos e a correspondente atribuição de significados são atribuições do pesquisador, assim como o foco principal de abordagem é o processo e seus significados.

- Quanto aos objetivos: em relação aos seus objetivos, uma pesquisa pode ser exploratória, descritiva ou explicativa (SILVA e MENEZES, 2000). Esta dissertação possui características de uma pesquisa explicativa, pois visa aprofundar o conhecimento sobre elementos envolvidos com a manufatura flexível e implementar um sistema supervisório que integre os diversos equipamentos existentes na célula.

- Quanto aos procedimentos técnicos: a pesquisa pode ser bibliográfica, documental, experimental, levantamento, estudo de caso, pesquisa *expos-facto*, pesquisa-ação ou pesquisa participante (SILVA e MENEZES, 2000). Esta dissertação contém pesquisa bibliográfica a respeito dos temas de integração e flexibilidade da manufatura, sistemas de manufatura, sistemas de supervisão e controle, redes de Petri, modelagem de sistemas de manufatura, padrão de comunicação OPC, *software* Elipse SCADA, hierarquias de sistemas de controle, FMS e FMC. As fontes de consulta utilizadas foram livros, revistas técnicas, anais de congressos, monografias, dissertações e teses, periódicos nacionais e internacionais, base de dados da CAPES e diversos *sites* na Internet. Em relação aos procedimentos técnicos, esta pesquisa também é experimental, pois manipula diretamente as variáveis relacionadas com o objeto de estudo (o laboratório da SOCIESC), ou seja, interfere na realidade procurando explicar os "porquês" (CERVO e BERVIAN, 1983). Ainda abordando os procedimentos técnicos, este trabalho pode também ser considerado como uma pesquisa-ação, pois foi concebido como resolução de um problema coletivo (a integração dos equipamentos do laboratório da SOCIESC para formação de uma FMC) e foi realizado com a participação cooperativa de outros trabalhos de pesquisa realizados no mesmo laboratório.

1.5.2 Pressupostos básicos

Os pressupostos básicos deste trabalho se referem aos elementos que fornecem um ponto de partida para o início do desenvolvimento do sistema supervisório, e da conseqüente integração da FMC, e numa visão geral podem ser resumidos a:

- Os Sistema de Manufatura Flexíveis envolvem um elevado grau de integração, onde o processamento de material é efetuado por centros de usinagem CNC dotados de trocadores de ferramentas e a transferência entre as máquinas é feita por veículos auto-guiados, ou esteiras automatizadas. O posicionamento e movimentação das peças são realizados por robôs. Os produtos em processamento e os acabados, além da matéria-prima são estocados em um armazém automático. Os recursos produtivos estão ligados a um sistema computacional que coordena todas as ações, capacitando o sistema a produzir diferentes tipos de peças, utilizando os mesmos equipamentos e o mesmo sistema de controle.

- Todas as máquinas e equipamentos do laboratório da SOCIESC são semelhantes aos equipamentos utilizados pela indústria, ou seja, não contam com recursos adicionais direcionados à integração.

- Os equipamentos existentes no laboratório da SOCIESC que são utilizados na formação da FMC são: um armazém automático, um robô industrial, um torno CNC e um centro de usinagem CNC. O número máximo de equipamentos CNC na célula será determinado pela capacidade do robô efetuar o carregamento e descarregamento destes CNCs, ou seja, é limitado pelo espaço de trabalho do robô.

- O conjunto de procedimentos e instruções, utilizados para realizar a integração física e lógica dos diferentes equipamentos presentes na célula, configuram um roteiro de implantação que pode ser utilizado para a construção de outras Células Flexíveis de Manufatura, compostas por um sistema de armazenamento automático, um robô industrial e um grupo de máquinas CNC.

- Os princípios aplicados na construção da FMC sustentam-se na idéia de que o sistema resultante deve ser modular, aberto, expansível, com ambiente distribuído e deve suportar equipamentos heterogêneos.

1.5.3 Roteiro da pesquisa

Os principais itens que compõem o roteiro da pesquisa são:

- Pesquisa bibliográfica a respeito dos principais tópicos necessários para integração da FMC: sistemas de manufatura, integração da manufatura, flexibilidade da manufatura, modelagem de sistemas de manufatura, Redes de Petri, sistemas de supervisão e controle, hierarquias de sistemas de controle, programação de CLPs (Moeller, Siemens e Mitsubishi), programação do robô ABB, redes de comunicação, padrão de comunicação OPC, *software* Elipse SCADA, programação de CNCs e banco de dados Access;

- Determinação de um conjunto mínimo de interfaces para cada equipamento do laboratório;

- Adaptações do *hardware* dos equipamentos CNC para dotá-los de funcionalidades necessárias para a integração (automatização das portas, dos dispositivos de fixação e da carga de programas de usinagem);

- Modelagem da FMC utilizando Redes de Petri Interpretadas;

- Construção dos gerenciadores, baseado na modelagem em RdPI, utilizando o *software* Elipse SCADA;

- Integração dos gerenciadores aos seus respectivos equipamentos, utilizando CLPs quando necessário;

- Integração dos diversos gerenciadores, através de uma rede de comunicação e utilizando a tecnologia OPC, para formação da FMC;

- Simulação da fabricação de um conjunto de peças para validação do trabalho.

1.6 DELIMITAÇÕES DO TRABALHO

Este trabalho não tem por objetivo discutir a necessidade ou não da implantação de FMSs e FMCs nas empresas que atuam na área da manufatura. Corrêa e Slack (1994), Chase et al. (2000) e Slack (2002) discorrem com muita propriedade sobre as necessidades, as vantagens e desvantagens das empresas flexibilizarem seus métodos produtivos. Este trabalho tem o objetivo de apresentar de forma detalhada uma alternativa para aquelas empresas, ou entidades de ensino, que queiram montar seus próprios Sistemas Flexíveis de Manufatura a partir de um sistema de armazenamento automático, um robô industrial, e até três equipamentos CNC.

A FMC resultante da integração dos equipamentos no laboratório da SOCIESC somente simula a fabricação de um conjunto de peças que compõem o jogo de xadrez (peão, torre, cavalo, bispo, dama e rei). A simulação segue todo o roteiro de fabricação normal de uma peça: cadastro do código da peça, cadastro da matéria-prima necessária para fabricação da peça, o armazém automático envia a peça para usinagem, o robô efetua o carregamento da matéria-prima no equipamento CNC correspondente, o CNC carrega o programa de simulação de usinagem, o CNC simula a usinagem da peça, o robô retira a peça pronta do CNC e a devolve ao armazém automático, que finaliza o processo disponibilizando a peça ao usuário. Para uma efetiva fabricação da peça, basta inserir o programa de usinagem na estrutura de programação do CNC.

A limitação da família de peças que a célula pode produzir é dada pela capacidade de transporte da garra do robô, pela capacidade de carga do *pallet* do armazém automático e pela capacidade de fixação e processamento dos equipamentos CNC.

A disposição física dos equipamentos no laboratório já estava previamente definida antes do início do projeto de pesquisa, e não permite mudanças significativas de *layout*. Os equipamentos produtivos são semelhantes aos encontrados na indústria, sem nenhum tipo de adaptação prévia visando à integração.

Todos os programas de movimentação do robô são residentes, assim como o programa de simulação de usinagem nos equipamentos CNC. Os gerenciadores não transferem programas aos equipamentos, apenas indicam qual programa residente deve ser executado. Todos os processos de usinagem devem ser cadastrados no banco de dados do gerenciador da célula, assim como a matéria-prima necessária para a fabricação das peças.

O gerenciador remoto permite que usuários fisicamente distantes insiram seus pedidos de fabricação na célula. Estes usuários devem estar dentro da rede local de computadores. O gerenciador remoto não permite que usuários insiram pedidos de fabricação utilizando a Internet.

O processo de validação deste trabalho é feito com a simulação da fabricação de todas as peças cadastradas no banco de dados, sendo que os pedidos de fabricação são realizados a partir do gerenciador FMC e do gerenciador remoto. Não é objetivo deste trabalho formular o melhor plano de processo ou sequenciamento de produção. Os roteiros de movimentação de materiais e de fabricação são pré-determinados pelo usuário, e o sequenciamento é determinado pela estratégia FIFO (*First In, First Out*), ou seja, o pedido mais antigo será atendido primeiro.

1.7 ESTRUTURA DA DISSERTAÇÃO

Esta dissertação é composta por seis capítulos, quatro apêndices e um anexo em CD-ROM. O primeiro capítulo realiza a introdução geral do trabalho, apresenta a metodologia e os objetivos gerais e específicos do projeto.

No capítulo 2 realiza-se a revisão bibliográfica. Definições sobre arquitetura e sistemas de fabricação mais comuns na atualidade são apresentadas, com especial ênfase nos Sistemas Flexíveis de Manufatura e na filosofia de manufatura integrada por computador. Os componentes físicos e lógicos dos sistemas de supervisão e controle também são desmembrados neste capítulo, assim como os principais recursos do *software* supervisório Elipse SCADA, que será utilizado para a construção dos gerenciadores da célula. As características do padrão de comunicação industrial OPC, utilizado para troca de informações entre os diferentes gerenciadores, também são detalhadas neste capítulo.

O terceiro capítulo trata da descrição geral do trabalho, apresentando o laboratório, os equipamentos e as condições de contorno do projeto de integração da célula. Particularidades da proposta são aprofundadas, assim como a revisão bibliográfica de alguns aspectos específicos que envolvem a formação da FMC.

A definição de um conjunto de interfaces para cada equipamento e a modelagem a eventos discretos da célula flexível de manufatura, utilizando o mecanismo formal de descrição das redes de Petri, compõe o quarto capítulo da dissertação.

O capítulo 5 relata em detalhes a construção dos gerenciadores da célula, a integração dos gerenciadores com os recursos produtivos através de CLPs, as alterações de *hardware* realizadas no torno e os programas desenvolvidos para CLPs, robô e *scripts* de programação do supervisório. A integração de todos os gerenciadores, por meio do padrão de comunicação OPC, também é descrita de forma pormenorizada.

As considerações finais, conclusões, contribuições do projeto e sugestões para continuidade do trabalho são apresentados no capítulo seis.

Os apêndices da dissertação contêm informações complementares sobre assuntos abordados nos diferentes capítulos, como: a totalidade das redes de Petri desenvolvidas para a FMC; os diagramas elétricos das modificações introduzidas no torno; os *scripts* de programação dos gerenciadores e os programas do robô. O anexo em CD ROM contém os aplicativos gerenciadores; o *software* Elipse SCADA; os *drivers*; os programas desenvolvidos para os CLPs Moeller e Siemens; o projeto da garra angular para o robô; as *tags* utilizadas, e a descrição das tabelas do banco de dados, no gerenciador AS/RS e FMC; os fluxogramas dos processos de entrada, saída e usinagem e o modelo de entidade/relacionamento no armazém; informações técnicas sobre o armazém automático, o torno CNC, o centro de usinagem CNC e do CLP Moeller.

2. REVISÃO BIBLIOGRÁFICA: HISTÓRICO, SISTEMAS, OPC E SUPERVISÓRIOS

Este capítulo realiza uma revisão bibliográfica a respeito dos principais tópicos que serão utilizados na construção da Célula Flexível de Manufatura. Um breve relato inicial, sobre o desenvolvimento dos sistemas de fabricação, mostra a evolução histórica dos sistemas até a atualidade. Também são descritos modelos de arquitetura para os sistemas de manufatura, e apresentados em detalhes os principais sistemas empregados atualmente, com especial ênfase aos Sistemas Flexíveis de Manufatura e à manufatura integrada por computador.

Sistemas de supervisão e controle, também conhecidos como sistemas SCADA, ou simplesmente supervisórios, são cada vez mais utilizados na automação de processos industriais. O sistema SCADA será um dos pilares da integração da Célula Flexível de Manufatura, e por isso também é abordado em detalhes neste capítulo.

Outro pilar para a integração da Célula Flexível de Manufatura é a comunicação entre os diferentes gerenciadores. Para cumprir esta função será utilizado o padrão de comunicação OPC, que permite que aplicações de supervisão e controle troquem dados entre si de forma aberta e simplificada.

Finalmente o *software* Elipse SCADA, utilizado na construção dos aplicativos gerenciadores da célula, tem suas principais características e funcionalidades analisadas.

2.1 HISTÓRICOS DOS SISTEMAS DE FABRICAÇÃO

Antes do advento da revolução industrial, toda a produção existente era fruto da ação manual de mestres, artesãos, artífices e aprendizes. Os produtos eram manufaturados, ou seja, eram realmente feitos à mão. A revolução industrial iniciou uma mudança radical nos processos de manufatura. Máquinas e ferramentas simples foram desenvolvidas, iniciando a mecanização nos processos produtivos. As primeiras fábricas localizavam-se próximas das fontes de energia e da matéria-prima. Inicialmente utilizou-se a energia hidráulica, razão pela qual muitas fábricas fixaram-se perto de rios. A força da água movimentava um eixo, que por sua vez movimentava as máquinas. Mais tarde, com a invenção da máquina a vapor, as fábricas ganharam flexibilidade quanto à sua localização (FAYET, 2002) e (FERREIRA, 1998).

A disposição física das máquinas foi herdada dos sistemas manuais de produção. As máquinas eram agrupadas de forma funcional, pela velocidade em que precisavam operar e pelo tipo de trabalho que executavam. Os processos eram arranjados conforme o grau de dificuldade e habilidades necessárias. Operadores desenvolviam habilidades específicas para trabalhar com cada tipo de máquina (CECCONELLO, 2002).

Com a invenção do motor elétrico, cada máquina ganhou autonomia de funcionamento. Isto possibilitou uma flexibilização a respeito da localização de cada máquina dentro das fábricas. Assim, puderam ser criados arranjos físicos funcionais que facilitassem a movimentação das peças durante o processo de fabricação. Surgiram assim os primeiros *layouts* dos novos sistemas de manufatura (FERREIRA, 1998) e (SLACK, 2002).

No início do século passado, Henry Ford idealizou a linha de montagem para fabricação de automóveis, o que possibilitou a produção seriada. Este novo método permitia fabricar grandes quantidades de um determinado produto, de forma padronizada, com o menor custo possível. A progressão do produto através do processo produtivo é planejada, ordenada e contínua. O trabalho vai até o operador em vez de o operador ir até o trabalho (MARTINS e LAUGENI, 1998).

A especialização da mão-de-obra, a divisão lógica e segmentada do trabalho e o aperfeiçoamento das máquinas-ferramenta, originaram a linha de montagem dedicada, uma extensão natural da linha de montagem de Henry Ford. Com este método de manufatura, produtos padronizados de alta qualidade são disponibilizados em grande quantidade a um preço acessível para o mercado consumidor. A montagem de um sistema de produção dedicado implica em grandes investimentos para as empresas. Devido à complexidade do sistema e da necessidade de recuperar os custos iniciais, mudanças nesta linha não são fáceis de serem implementadas. Quando o mercado ficava saturado de produtos padronizados, os fabricantes introduziam pequenas modificações e variantes nos produtos. Isto trouxe naturalmente a necessidade de incorporar certo grau de flexibilidade nos sistemas de manufatura em massa. A invenção do transistor foi particularmente importante para a flexibilização da manufatura, pois possibilitou que os controladores das máquinas ganhassem certa capacidade de reprogramação e aumentou o número de produtos que elas podiam fabricar (CECCONELLO, 2002) e (FERREIRA, 1998).

Na década de 40 ocorreu um grande avanço na automação programável, com a invenção do controle numérico (CN) nos laboratórios do MIT (Massachusetts Institute of Technology). A nova tecnologia permitiu a usinagem de peças complexas em pequena escala, fornecendo uma elevada flexibilidade aos sistemas de manufatura. O CN foi a primeira unidade de controle de máquinas que

aliou a capacidade de programação com o *hardware*. A integração do transistor na forma de circuitos integrados e o surgimento dos microcomputadores trouxe uma grande capacidade de adaptação e controle para os processos de manufatura. A tecnologia de automação convencional já não tinha a capacidade de responder às novas e crescentes necessidades do mercado consumidor. As possibilidades de combinação de *hardwares* miniaturizados, com grande capacidade de processamento, e *softwares* de controle, tornaram as fábricas mais flexíveis e ágeis nos seus processos de fabricação (FERREIRA, 1998).

Os constantes avanços tecnológicos na área de *hardware* e *software*, aliados às demandas de uma sociedade que exige constantemente produtos novos, acessíveis, personalizados e com qualidade, abrem as portas para o desenvolvimento de métodos produtivos cada vez mais elaborados e complexos.

2.2 ARQUITETURA PARA SISTEMAS DE MANUFATURA

A arquitetura de um sistema de manufatura é o resultado do projeto do sistema, onde são especificadas as funções dos componentes, suas interfaces, interações e restrições. Ela tem a função de descrever de forma geral e resumida, a complexidade dinâmica de um sistema a partir de modelos mais simples (WYNS et al, apud BATOCCHIO e FIORONI, 2006). A definição de uma arquitetura fornece uma abstração do sistema complexo de forma simples e auxilia o projetista a definir as interfaces e interações entre os diversos recursos produtivos. Ela também permite identificar as áreas e componentes mais importantes para o sistema, minimizando os impactos de eventuais necessidades de mudanças ao apontar onde o processo pode ser modificado, de forma tópica, e quais componentes não podem ser alterados quando da adaptação do sistema para outro uso.

Alguns dos principais benefícios concedidos pela adoção de uma arquitetura de sistema de manufatura são: estabelecimento de uma terminologia unificada, sem ambigüidades e que seja conhecida por todos os envolvidos; simplicidade no projeto do sistema, facilitando o desenvolvimento de sua arquitetura; maior qualidade e robustez no desenvolvimento dos sistemas, por estarem baseados em conceitos confiáveis e comprovados pela arquitetura; interfaceamento e possibilidade de reaproveitamento de módulos de arquitetura em diferentes projetos ou gerações de sistemas; fácil identificação das soluções utilizadas, pois a arquitetura precisa indicar e justificar de forma clara e

concisa quando e como cada estágio do desenvolvimento recebeu implementações de engenharia (BATOCCHIO e FIORONI, 2006).

Os sistemas de manufatura existem desde o momento em que o homem começou a utilizar ferramentas que o auxiliavam a realizar tarefas. Entre os sistemas de manufatura mais utilizados na atualidade estão o sistema funcional, o sistema em linha, o sistema celular e os sistemas flexíveis. A demanda imposta pelos anseios do mercado faz com que a proposta e o desenvolvimento de novos sistemas de produção sejam incessantes. Entre essas novas propostas destacam-se o sistema de manufatura ágil e o sistema holônico de manufatura.

2.2.1 Sistema Funcional

O sistema funcional, também conhecido por *job shop*, caracteriza-se por agrupar os recursos fabris da empresa, que desempenham a mesma função, em setores. A matéria-prima transita entre os diferentes setores enquanto vai sendo processada.

Para a produção de cada tipo de peça no *job shop* são estabelecidos roteiros que indicam a seqüência de máquinas que serão utilizadas para processar o pedido. As peças são movimentadas entre os diferentes setores com a ajuda de pequenos carros ou empilhadeiras. A figura 2.1 apresenta o *layout* típico de um sistema *job shop*.

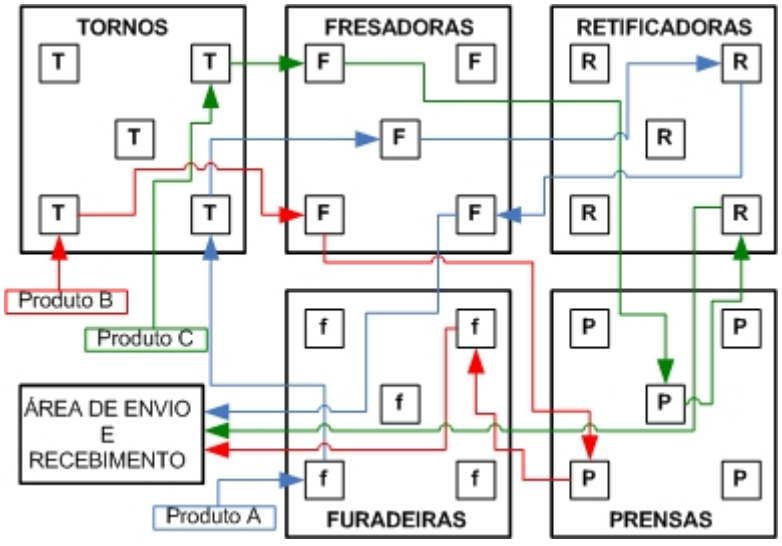


Figura 2.1. *Layout* típico do sistema funcional.

O *job shop* apresenta grande flexibilidade de produção. Qualquer operação pode ser feita em qualquer ordem. O sistema é robusto, pois a eventual quebra de uma máquina dificilmente ocasiona a

parada de todo o sistema, e o agrupamento de funções facilita o treinamento dos funcionários. Entre as características negativas do *job shop*, pode-se destacar o alto tempo de fabricação devido ao *setup* e movimentação, grande complexidade de roteamento, grande quantidade de estoque intermediário, baixo tempo produtivo, dificuldade na manutenção da qualidade do produto e alto custo de produção.

O sistema funcional é muito utilizado para fabricação de lotes pequenos, produtos muito diversificados ou que sofrem constantes mudanças, e prazos de entrega curtos (FERREIRA, 1998), (SLACK, 2002), (CECCONELLO, 2002) (MARTINS e LAUGENI, 1998).

2.2.2 Sistema em Linha

O sistema em linha, também conhecido como *flow shop*, é um *layout* utilizado nas linhas de montagem, ou seja, de produção em massa, e é formado em função das necessidades específicas do produto a ser fabricado.

No *flow shop* os diferentes recursos produtivos estão posicionados na seqüência das operações exigidas para a fabricação do produto. Desta forma o sistema apresenta boa velocidade de fabricação, permitindo atender rapidamente ao pedido de grandes lotes de um mesmo produto. A figura 2.2 apresenta um diagrama em blocos típico do *layout flow shop.*

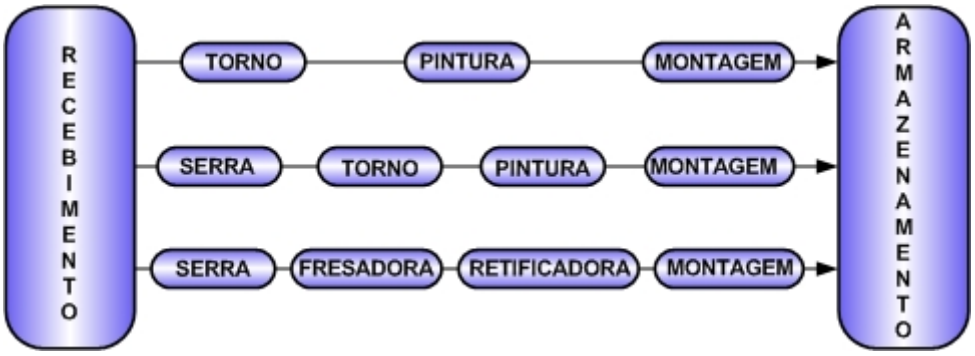


Figura 2.2. *Layout* típico de um sistema em linha.
(Fonte: adaptado de FERREIRA, 1998)

A flexibilidade dos sistemas em linha é muito baixa, e o seu custo de implementação normalmente é muito alto, o que o torna viável somente nos casos em que a empresa possui vários produtos similares e com tempo de vida elevado. Outro problema reside no fato de que uma única máquina quebrada pode interromper completamente a produção (FERREIRA, 1998), (SLACK, 2002), (BATOCCHIO e FIORONI, 2006), (MARTINS e LAUGENI, 1998).

2.2.3 Sistema Celular

O sistema celular pode ser utilizado na fabricação de pequenos lotes de peças que apresentam similaridades entre si. Este sistema é aplicado atualmente na manufatura enxuta (*Lean Manufacturing*). A característica marcante do sistema é a de agrupar todas as máquinas utilizadas para a fabricação de um grupo – ou família de peças – formando uma célula. As peças poderão ser processadas por algumas, ou todas as máquinas da célula. A figura 2.3 mostra uma célula de fabricação, que normalmente possui um formato em "U".

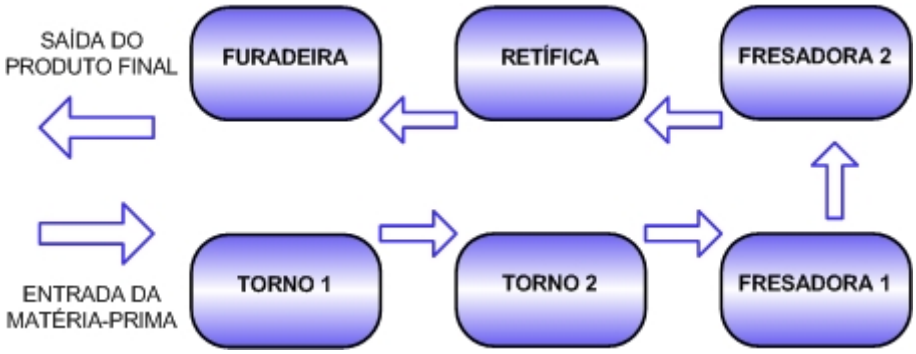


Figura 2.3. *Layout* de um sistema celular.

As principais vantagens do sistema celular são: permitir a fabricação de lotes pequenos; reduzir os estoques intermediários; melhorar os índices de aproveitamento de equipamentos e mão de obra; permitir a manutenção de altos níveis de qualidade. Entre as desvantagens apresentadas pelo sistema está a dependência em relação ao bom funcionamento das máquinas. Qualquer imprevisto, como manutenção ou quebra, pode paralisar toda a célula. A flexibilidade é também relativamente baixa, pois está restrita a um grupo de peças. Se houver a necessidade de introduzir uma mudança significativa no desenho da peça a ser fabricada, a célula poderá não ser capaz de produzi-la (FERREIRA, 1998), (AGUIAR, 2003), (BATOCCHIO e FIORONI, 2006).

2.2.4 Sistemas Flexíveis de Manufatura

Uma definição geral de Sistema Flexível de Manufatura (FMS, na sigla em inglês), apresentada por Kaltwasser (*in* SEVERIANO FILHO, 1995), sustenta a idéia de que os FMSs são

sistemas de produção altamente automatizados, capazes de produzir uma grande variedade de diferentes peças e produtos usando o mesmo equipamento e o mesmo sistema de controle. Já Slack et al. (2002) afirma que os FMSs combinam diferentes tecnologias em um sistema único, e apresenta a seguinte definição: "uma configuração controlada por computador de estações de trabalho semi-independentes, conectadas por manuseio de materiais e carregamento de máquinas automatizadas".

É comum encontrar na literatura grandes variações sobre a definição de FMS. Muitos aspectos deste sistema ainda estão em discussão. Segundo Sanches (2006), sob uma determinada ótica, os FMSs podem ser consideradas como um conjunto interligado de Células Flexíveis de Manufatura (FMCs). Por outro lado, um grupo de máquinas dedicadas a um único propósito, dotado de flexibilidade de fluxo de materiais entre as estações e podendo utilizar diferentes combinações de máquinas para processar a matéria-prima, também é considerado um FMS por alguns especialistas.

O conceito de FMS envolve um alto nível de automação com mínima necessidade de intervenção de operadores. A fabricação é realizada por centros de usinagem automatizados e a transferência entre as máquinas é feita por AGVs (Veículos Auto-Guiados) ou esteiras automatizadas. O posicionamento e movimentação das peças são realizados por robôs ou similares. Os produtos já processados, ou em fase intermediária do processo, e a matéria-prima são armazenados em um armazém automático. Todos os componentes estão ligados a um sistema computacional que coordena todas as ações do sistema. A figura 2.4 apresenta o exemplo de um FMS comercial.

Figura 2.4. Exemplo de um FMS.
(Fonte: http://www.heller-machinetools.com/bra/sites/fms.htm)

MacCarthy e Liu (1993) sugeriram uma classificação abrangente para os FMSs. Esta proposta baseia-se em fatores como nível de flexibilidade, grau de robotização e automação, configuração e capacidade operacional, *layout* e níveis de controle. A classificação proposta pelos autores abarca de forma geral a existência de três subsistemas nos FMSs: processamento, manuseio e armazenamento de material, e controle computadorizado.

- Sistema de processamento - Este sistema é formado a partir de um grupo de máquinas de comando numérico computadorizado (CNC), com capacidade para troca de ferramentas. Esses elementos capacitam o FMS a processar, ao mesmo tempo, diferentes tipos de produtos, atribuindo-lhe flexibilidade;
- Sistema para manuseio e armazenamento de material - Este sistema é destinado à provisão e gerenciamento do material, a partir de equipamentos robotizados e/ou automatizados. Através desses elementos o sistema adquire flexibilidade de movimentação;
- Sistema de controle computadorizado - Este sistema é encarregado de efetuar o controle operacional do conjunto do sistema de manufatura.

Alguns dos benefícios relatados por usuários de FMS são (SLACK et al, 2002):

- Redução do *lead time* e do tempo de atravessamento (porta-a-porta da fábrica) entre 60 e 70%;
- Economia de estoque (especialmente de material em processo) e fluxo de materiais mais uniforme ao longo da fábrica, com menos formação de filas de materiais esperando usinagem;
- Utilização aumentada (nas manufaturas por lotes, o nível de utilização de equipamento é relativamente baixo, já que muito tempo é gasto esperando que produtos sejam colocados nas máquinas). Melhorias na faixa de 200 a 400%;
- Redução do tempo de preparação (associada à melhoria dos níveis de utilização). Melhoramentos relatados entre 50 e 90%;
- Número de máquinas ou operações reduzido (derivado da integração física das operações nas máquinas);
- Qualidade aumentada (não somente atribuído ao FMS); melhoramentos encontram-se na faixa de 20 a 90%;
- Economia de espaço, redução da dependência de subcontratados, economia no uso de mão de obra especializada, prontidão de resposta aos consumidores aumentada (rapidez e qualidade de

serviço), ciclos de inovação da produção mais rápidos e capacidade melhorada de fazer protótipos.

Entre os aspectos negativos dos FMS, o principal, sem dúvida, é o alto custo de implantação. O grau de automação e a complexidade do sistema de controle dos FMSs limitam a possibilidade de implantação deste sistema somente a empresas com grande capacidade financeira. Um resumo das principais características dos Sistemas Flexíveis de Manufatura é descrito por Ferreira (1998):

> "Tais sistemas são caros para projetar, e são complexos de analisar e controlar.... Quase todos estes sistemas são encontrados em empresas bem grandes que podem comprá-los, ou que receberam suporte financeiro governamental (militar, defesa). O FMS apresenta a filosofia da super-máquina por excelência. Fundamentalmente, pretende-se combinar a flexibilidade de um *job shop* com a produtividade das linhas dedicadas....O computador que controla o FMS deve lidar com a(s) esteira(s), manter a biblioteca de programas NC e carregá-los nas máquinas, manusear o agendamento do FMS, rastrear a manutenção das ferramentas, rastrear o desempenho do sistema e imprimir os relatórios de gerenciamento. Não há nenhuma surpresa no fato de que freqüentemente o *software* de FMSs é o fator que mais limita tais sistemas".

Uma Célula Flexível de Manufatura (FMC) pode ser considerada como um módulo do Sistema Flexível de Manufatura (FMS). Na FMC ocorre efetivamente a fabricação das peças; é a unidade básica de produção formada a partir da integração de máquinas de processamento de material (CNCs), de dispositivos de manipulação e movimentação de peças (AGVs, robôs e esteiras), *buffers* de matéria-prima e peças (armazém automático e esteiras) e um sistema computacional.

2.2.5 Manufatura Integrada por Computador

A filosofia da manufatura integrada por computador (CIM, na sigla em inglês), preconiza a utilização da tecnologia da informação para integrar as funções técnicas, operacionais e de produção nas empresas. O uso cada vez maior de computadores nas empresas e o desenvolvimento de *softwares* para CAE (*Computer Aided Engineering*), CAD (*Computer Aided Design*), CAM (*Computer Aided*

Manufacturing) e CAPP (*Computer Aided Process Planning*) permitiu a integração de diversos departamentos através de redes de comunicação.

Segundo Vieira (1996), CIM é a integração de todas as atividades envolvidas na manufatura (por exemplo, vendas, compras, projeto, planejamento, administração, finanças e produção) através de uma rede de comunicação e de um *software* de gerenciamento (integrador CIM), com o objetivo de melhorar a eficiência organizacional, pessoal e de produção.

Em um ambiente de manufatura totalmente integrada, o ciclo de produção tem início com o pedido do cliente ao departamento de vendas, quando o mesmo fornece as especificações do produto desejado, quantidade, prazo de entrega etc. Essas informações são repassadas para a engenharia de projetos, onde o projetista avalia as especificações do produto. Com o auxílio do CAD, o produto é inicialmente representado em um modelo geométrico e posteriormente desmembrado em desenhos e lista de materiais, que são então encaminhados à engenharia de fabricação.

A partir dos modelos geométricos gerados pelo CAD, o CAPP elabora um roteiro (plano de processo) que torne possível a fabricação do produto, considerando as características dos recursos produtivos. O plano de processo é finalmente repassado para o setor de planejamento e controle da produção que, com o auxílio do CAM, verifica a disponibilidade de máquinas, equipamentos e ferramentas, e gera os parâmetros para o gerenciamento e efetiva fabricação do produto desejado.

A figura 2.5 exemplifica o funcionamento de uma arquitetura CIM.

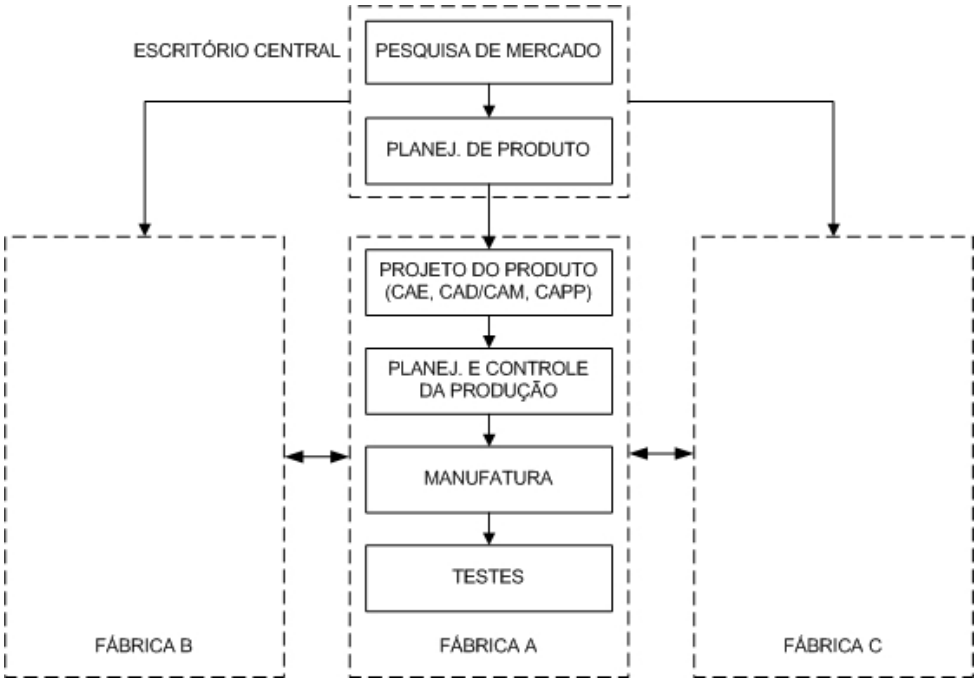


Figura 2.5. Funcionamento de um sistema CIM.
(Fonte: adaptado de BATOCCHIO e FIORONI, 2006)

A utilização do CIM permite aumentar a eficiência produtiva da empresa, pois reduz a redundância de informações e a interferência do operador. Ela também permite melhoria dos serviços prestados aos clientes, melhoria na qualidade do produto, redução de estoques e de estoques intermediários (WIP), redução do *lead time* e aumento da flexibilidade e da produtividade (REMBOLD et al, 1993).

Um resumo macro do CIM pode ser visto na representação formal dada pelo diagrama entidade/relacionamento da figura 2.6. Neste diagrama, as tecnologias concorrentes que dão sustentação ao CIM (monitoração e controle, redes industriais, computadores e *softwares*, banco de dados e equipamentos) são apresentadas. O desmembramento dos diferentes tipos de equipamentos, com ênfase naqueles utilizados nos processos de fabricação discretos, possibilita visualizar a posição das FMCs e FMSs em relação ao CIM.

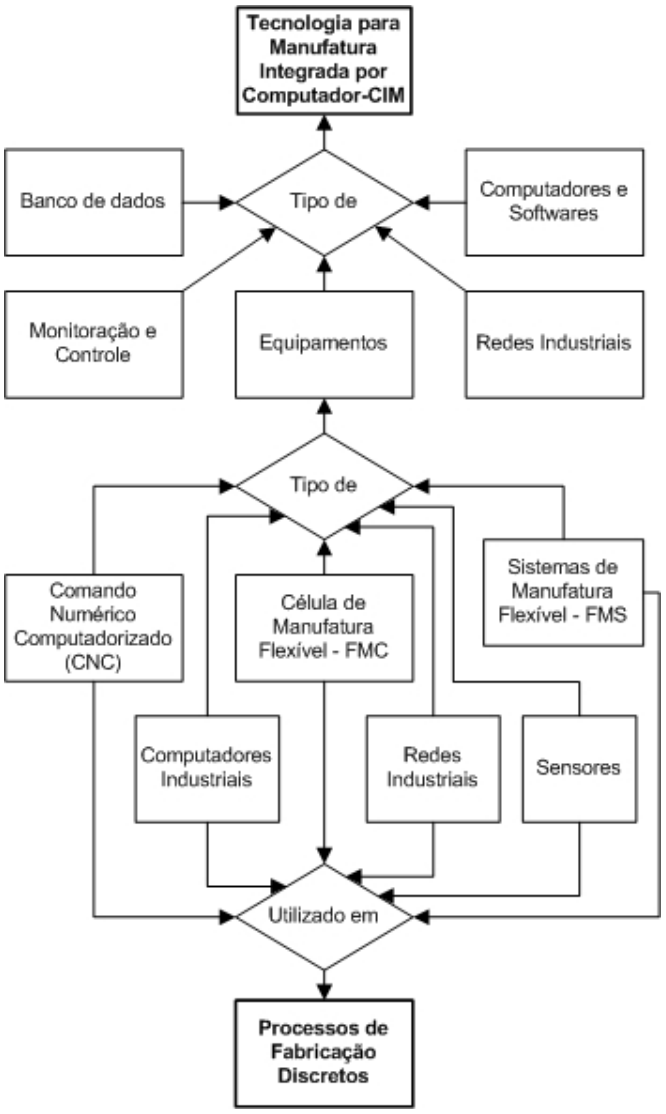


Figura 2.6. Diagrama entidade/relacionamento da tecnologia do CIM.
(Fonte: adaptado de DE NEGRI, 2004)

2.2.6 Comparativo entre os sistemas de manufatura

Os diferentes sistemas de manufatura abordados anteriormente possuem características que os diferenciam em relação a volume de produção, custo, flexibilidade, etc. A tabela 2.1 apresenta um quadro comparativo das características principais dos sistemas, para diferentes volumes de produção.

Tabela 2.1. Comparativo entre sistemas para diferentes volumes de produção.
(Fonte: Adaptado de SANCHES, 2006)

	Produção de peças	Produção em lotes	Produção em massa
Volume anual de produção	1 a 10.000	5.000 a 200.000	Mais de 100.000
Motivação principal	Capacidade	Flexibilidade	Volume
Custo unitário	Muito alto	Baixando	Mínimo
Ferramenta de corte	Padrão	Algumas especiais	Personalizadas
Manuseio automatizado das peças	Raro	Às vezes	Sempre
Flexibilidade para fabricar peças:			
Totalmente diferentes	Sim	Possível	Impossível
Similares, com poucas diferenças	Sim	Sim, se estava previsto	Muito limitada
Possibilidade de troca de materiais	Sim	Limitada	Extremamente limitada
Possibilidade de mudança gradual	Sim	Possível	Difícil
Máquina-ferramenta recomendada	CNC	Centro CNC, FMC e FMS	Sistema em linha
Aplicações típicas	Aviação, moldes e ferramentas	Agricultura, motores e máquinas	Indústria automobilística

A escolha de um sistema de produção na indústria está intimamente ligada ao tipo e natureza de seus produtos. São eles que determinarão a filosofia de produção, desde o *layout* ao tipo de máquinas que serão utilizadas, passando pelo tamanho de lote ideal, política de estoques, flexibilidade, roteamento, logística de distribuição, etc.

Os sistemas que se caracterizam por grandes taxas de produção, normalmente não possuem flexibilidade para fabricar diferentes tipos de peças. Já os sistemas que apresentam grande flexibilidade, não conseguem produzir lotes médios com preços econômicos. Os FMSs incorporam algumas das características de ambos sistemas, como a alta taxa de utilização dos recursos produtivos, qualidade e flexibilidade, porém para volumes médios de produção. Em países industrializados, o nível de participação dos produtos feitos em massa é de 25% do total. Nesses mesmos países, os produtos fabricados em volumes médios de produção representam 40% do total (CHASE et al, 2000). Nos Estados Unidos, por exemplo, 75% dos produtos eram fabricados em lotes de no máximo 50 unidades, segundo Lorini (1993) e Zhou e Venkatesh (2000). A figura 2.7 apresenta um quadro

comparativo entre a taxa de produção típica dos diversos sistemas analisados e a variedade de peças que estes sistemas podem produzir.

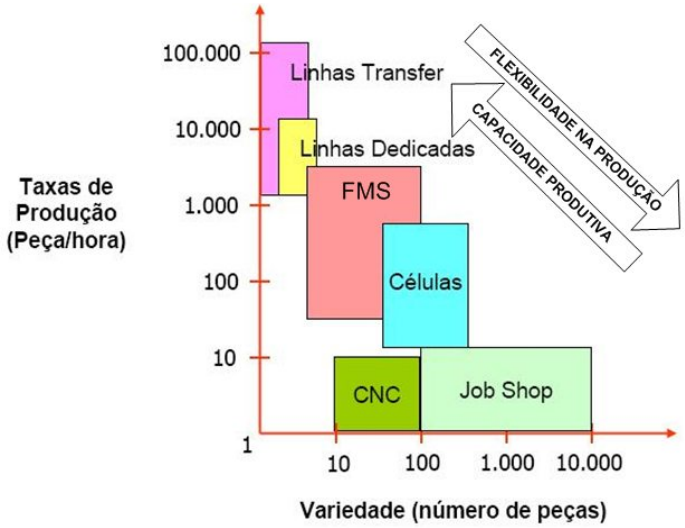


Figura 2.7. Características de capacidade produtiva e flexibilidade dos sistemas de produção.

2.3. SISTEMAS DE SUPERVISÃO E CONTROLE

Os sistemas de supervisão e controle são sistemas desenvolvidos para funcionar como interfaces homem-máquina, estações de supervisão local de processos industriais ou estações concentradoras de dados em processos distribuídos. Estes sistemas são baseados em microcomputadores interligados a controladores programáveis, estações remotas ou outros equipamentos de aquisição de dados (SEIXAS FILHO, 1999).

Os sistemas supervisórios também são chamados de SCADA (*Supervisory Control and Data Aquisition*). Daneels e Salter (2000) explicam que os *softwares* supervisórios permitem que sejam monitoradas e rastreadas informações de um processo produtivo ou uma instalação física. Estas informações são coletadas por meio de equipamentos de aquisição de dados, manipuladas, analisadas, decompostas, armazenadas e exibidas ao usuário. Pires et al. (2005), ao analisar o grau de importância dos sistemas de supervisão e controle para a indústria, afirma que:

> "Os sistemas SCADA são de importância estratégica já que são adotados na maioria das indústrias que compõem a infra-estrutura de um país. As aplicações da tecnologia

> SCADA alcançam praticamente todo o espectro do setor produtivo. Para exemplificar, esses sistemas são utilizados na indústria química, petroquímica e de cimentos; na indústria alimentícia; na produção e distribuição de energia elétrica; na distribuição de água; no controle de oleodutos, gasodutos, centrais nucleares, edifícios inteligentes e tráfego".

No início os sistemas supervisórios eram arquiteturas centralizadas, fechadas e sem conectividade externa, que utilizavam *hardware* e *software* proprietários. As informações eram visualizadas em um painel de lâmpadas e indicadores, sem qualquer interface aplicacional com o operador. Atualmente, os sistemas de manufatura utilizam tecnologias de computação e de comunicação que permitem automatizar a monitoração e controle de processos industriais. Variáveis numéricas e alfanuméricas são coletadas dos processos, às vezes geograficamente distantes, e disponibilizadas ao operador de forma amigável, com recursos gráficos e interatividade. Estas variáveis recebem o nome de "*tags*" e representam pontos de entrada e saída do processo que está sendo controlado (SILVA e SALVADOR, 2005).

Existem atualmente no mercado vários *softwares* supervisórios, dentre estes, Barros (2005) apresenta a descrição detalhada das principais características do supervisório Legato, da Gefasoft, que gerencia as células automatizadas da linha de produção de uma das plantas da Volkswagen do Brasil. Gaidzinski (2003) realizou uma pesquisa comparativa entre alguns supervisórios disponíveis no mercado, dentre os quais o Axeda Supervisor e Wizcon, da Axeda Systems Inc; iFIX, da Intellution; Elipse SCADA e Elipse E3, da Elipse; Indusoft, da Indusoft e InTouch, da Wonderware.

No processo de escolha do supervisório para a construção dos gerenciadores da célula, além dos *softwares* anteriormente mencionados, também foram pesquisados os seguintes produtos SCADA disponíveis no mercado:

- Factory Link 7, fornecido pela Usdata (http://www.usdata.com/). É um supervisório orientado a objeto desenvolvido para aquisição e controle de dados críticos em chão-de-fábrica. Possui uma ampla biblioteca com várias funcionalidades típicas de um ambiente de manufatura, facilitando o desenvolvimento de aplicativos. Utiliza a tecnologia OPC para aquisição e distribuição de dados.
- Paradym-31, da Advantech (http://www.advantech.com/). Provê um ambiente gráfico de programação compatível com o MS Windows. Também é compatível com a norma IEC 1131-3, o que redunda em diminuição dos custos de programação e treinamento.

- WizFactory, da eMation (http://www.emation.com/). Apresenta uma solução completa para automação. Possui módulos que permitem interagir com os processos via Internet, utilizando um navegador padrão. Também emula o comportamento lógico de um CLP e fornece ferramentas para controle da planta em tempo real.
- Cimplicity, da GE Fanuc (http://www.gefanuc.com/). È um supervisório orientado a objeto que proporciona informações confiáveis e em tempo real do chão-de-fábrica. Incorpora tecnologias como OBDC (*Open Data Base Connectivity*), ActiveX e OPC. É utilizado em todas as unidades da General Electric como uma norma corporativa para monitorar os processos de melhoria baseados em Seis Sigma. Possui funções que permitem a conexão com base de dados relacionais (Access, SQL Server e Oracle), assim como com sistemas ERP (*Enterprise Resource Planning*), como o SAP.
- Genesis32, da Iconics (http://www.iconics.com/). É um sistema SCADA orientado a objeto que incorpora tecnologias como OPC, ActiveX e programação em VBA (Visual Basic). Interage com facilidade com banco de dados e permite controle de processos via Internet.
- Intellution Dynamics, da Intellution (http://www.intellution.com/). É uma família de *softwares* SCADA para automação que constitui uma das soluções mais poderosas disponíveis no mercado. Possui módulos IHM (Interface Homem-Máquina), de controle de processos em batelada, simulação de CLPs virtuais e aplicações de supervisão e controle via Internet.
- LabView, da National Instruments (http://www.ni.com/). O LabView oferece um ambiente de desenvolvimento gráfico fácil de usar e voltado para aplicações científicas e de engenharia. Permite criar interfaces para instrumentação virtual (osciloscópios, geradores de função, etc.), sem a necessidade de gerar códigos de programação. Também apresenta um grande número de cartões para aquisição de dados de campo e facilidade para construir aplicativos em um ambiente totalmente gráfico, que podem ser executados em rede ou pela Internet.
- HMI/SCADA Paragon, da Nematron (http://www.nematron.com/). É um *software* SCADA poderoso e ao mesmo tempo flexível que incorpora tecnologias Activex, OPC, COM (*Component Object Model*) e DCOM (*Distributed Component Object Model*). A mesma base de dados criada para o sistema de controle é utilizada para configurar I/Os, telas de operador e aquisição de dados, o que facilita a programação e a depuração de erros.
- FactoryFloor Software, da Opto 22 (http://www.opto22.com/). É um *software* composto por uma série de módulos para automação e controle. O módulo OptoControl oferece um ambiente gráfico e intuitivo de programação que combina controle analógico, digital,

comunicação serial e de rede. OptoDisplay é o módulo IHM, com recursos multimídias. OptoServer é o servidor de dados OPC e OptoConnect proporciona uma interface bidirecional entre a base de dados e os sistemas de controle.

- RSView32, da Rockwell Automation (http://www.software.rockwell.com/). É um *software* construído para monitorar e controlar máquinas automatizadas e processos, e que opera em ambiente MS Windows. É totalmente compatível com a tecnologia OPC, o que facilita a comunicação com um grande número de dispositivos de *hardware*. Permite programação em VBA e supervisão e controle de processos pela Internet.
- HYBREX (Hybrid Expert System), WinCC HMI, Web Control Center (webCC), SIMATIC WinAC ODK (Open Developer Kit), SIMATIC WinAC (Windows Automation Center), da Siemens (http://www.siemens.com.br). A Siemens oferece um grande número de *softwares* para simulação de mudanças de processos, integração de *softwares* na manufatura com base de dados, controle e supervisão de processos pela Internet, ferramentas para desenvolver interfaces para transmissão de dados em tempo real, automação e integração da manufatura, desenvolvimento de IHMs, simulação de programas de CLPs, etc.
- Aimax, da TA-Engineering Products (http://www.ta-eng.com/home.htm). É um supervisório robusto e poderoso que opera na plataforma Windows. É capaz de armazenar e integrar dados proveniente de diversas fontes, graças a um grande número de interfaces para CLPs, controladores e dispositivos de I/O. Fácil de usar, conta com uma grande biblioteca de símbolos que facilitam a construção de aplicativos. Possui uma base de dados relacional proprietária que melhora o desempenho e a flexibilidade na manipulação de dados.
- FactorySuite 2000, da Wonderware (http://www.wonderware.com/). É composto por um conjunto de módulos de *software* industrial voltados ao controle de processos. O módulo Intouch fornece, através de uma interface gráfica, a visualização centralizada de todos os recursos de controle e informações do chão-de-fábrica. O módulo InControl é um sistema de controle de arquitetura aberta que permite projetar, criar, testar e executar aplicativos para controle de processos. O módulo IndustrialSQL Server é uma base de dados relacional combinada com um sistema de controle de tempo real.
- SuiteVoyager, da Wonderware (http://www.wonderware.com/). Incorpora os módulos Intouch, IndustrialSQL e I/O Servers do FactorySuite, e permite que o usuário tenha acesso aos dados através da Internet. O supervisório é totalmente compatível com a tecnologia XML.

Para construção dos diversos gerenciadores da FMC no laboratório da SOCIESC foi escolhido o *software* Elipse SCADA. São várias as razões que levaram a esta decisão, entre elas:

- Porque o *software* apresenta todas as funcionalidades necessárias para a construção do supervisório, como IHM (Interface Homem-Máquina), ODBC (*Open Data Base Connectivity*) que permite troca de dados com a base de dados Access, incorpora a tecnologia OPC para aquisição de dados de CLPs e troca de informações com outros aplicativos SCADA em tempo real, controla e monitora todos os processos no chão-de-fábrica, permite emitir relatórios formatados, históricos, controle de alarmes e Controle Estatístico do Processo (CEP);
- Porque é um *software* nacional, de reconhecida trajetória e utilizado por empresas como Michelin, General Motors, Mercedes Benz, Fiat, Vokswagen, Ford, Dana Albarus, Petrobrás, Stihl, Sadia, Ceval, Perdigão, Nestlé, Batavo, Gerdau, CSN, Cia. Vale do Rio Doce, Usiminas, Sabesp, Copel e por estar presente em paises como Alemanha, Estados Unidos, Índia, Taiwan, Rússia, Bielo-Rússia, Malásia, Portugal, Suécia, Porto Rico, Argentina, Colômbia e Chile, entre outros (http://www.elipse.com.br);
- Por ser um *software* com boa documentação e com suporte técnico disponível;
- Pela experiência prévia do pesquisador com o supervisório e porque o laboratório da SOCIESC conta com a autorização de uso necessária (*hardkey*).

As principais características da estrutura do supervisório Elipse SCADA são apresentadas em detalhes no item 2.6 deste capítulo, assim como particularidades dos objetos de tela, tipos de *tags*, programação de *scripts*, construção de aplicativos, *drivers* de comunicação e tecnologia OPC.

2.3.1. Componentes Físicos de um Sistema SCADA

Os componentes físicos principais de um sistema de supervisão são: sensores, transdutores e atuadores; estações remotas de aquisição; rede de comunicação; controle e monitoração central, representado pelo sistema computacional SCADA.

Os sensores e transdutores são os responsáveis pela aquisição e conversão primária de dados em campo. Grandezas físicas como temperatura, pressão e velocidade, assim como presença de uma

peça na esteira, ou a porta aberta de um torno CNC, são convertidos em sinais digitais ou analógicos para as estações remotas. Os atuadores têm a função de atuar sobre o sistema monitorado, basicamente ligando ou desligando dispositivos.

As estações remotas de aquisição normalmente são compostas por CLPs (Controladores Lógicos Programáveis) ou RTUs (*Remote Terminal Units*), unidades computacionais robustas utilizadas em processos industriais, que têm a função de processar os dados recebidos dos elementos em campo, efetuar cálculos ou ações de controle e atualizar saídas (DANEELS e SALTER, 2000).

A rede de computação é o meio por onde circulam as informações das estações remotas para o sistema SCADA, assim como entre diferentes sistemas SCADA. Elas podem ser implementadas por meio de cabos metálicos, fibras óticas, *links* de rádio, etc (DAYTON-KNIGHT, 2006).

A principal unidade dos sistemas SCADA é a estação de monitoração central, ou MTU (*Master Terminal Unit*). Ela é a responsável por receber as informações das estações remotas, processá-las e interferir nos processos. A estação de monitoração central pode estar centralizada em um único computador ou distribuída em uma rede de computadores, compartilhando as informações coletadas. A figura 2.8 apresenta um resumo da estrutura física de um sistema SCADA.

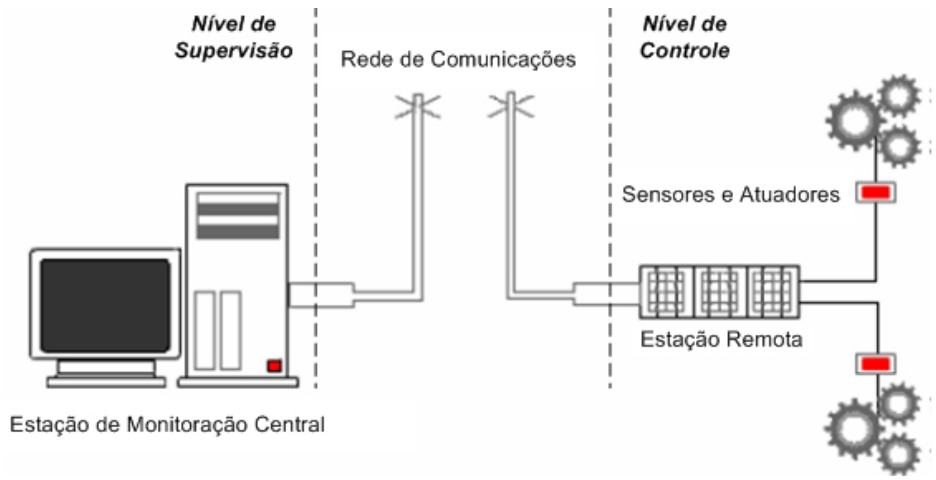


Figura 2.8. Sistema de Supervisão e Controle.
(Fonte: adaptado de SILVA e SALVADOR, 2005)

2.3.2 Componentes Lógicos de um Sistema SCADA

Geralmente os sistemas SCADA dividem suas principais tarefas internas em módulos, que permitem uma maior ou menor conectividade, flexibilidade e robustez, dependendo da aplicação e da

necessidade do sistema, segundo Silva e Salvador (2005). Numa análise macro, essas tarefas podem ser divididas em: núcleo de processamento, comunicação com CLPs e RTUs, gerenciamento de alarmes, histórico e banco de dados, lógica de programação interna (*scripts*) ou controle, interface gráfica, relatórios, comunicação com outras estações SCADA e comunicação com sistemas externos.

As informações coletadas em campo são enviadas para o núcleo principal do *software*, que é o responsável por distribuir e coordenar o fluxo destas informações para os outros módulos. Cada módulo processa as informações pertinentes até que as mesmas estejam disponíveis ao operador do sistema, na interface gráfica ou console de operação do processo. Através do núcleo principal, o operador tem acesso à representação gráfica, na forma de ícones, dos processos que estão sendo supervisionados ou controlados (DAYTON-KNIGHT, 2006). As informações podem ser apresentadas ao operador também sob a forma de gráficos, relatórios, animações, alarmes visuais e auditivos.

2.4 PADRÃO "OLE for PROCESS CONTROL" (OPC)

O padrão de comunicação OPC foi escolhido neste trabalho para integrar os diversos gerenciadores pelas seguintes razões: (a) A primeira delas é que a tecnologia OPC permite a troca de dados entre aplicativos, e entre aplicativos e CLPs, em tempo real e sem a necessidade de uso de *drivers* proprietários, que é uma das propostas iniciais deste projeto de pesquisa. (b) A segunda razão está no fato de que o *software* de supervisão e controle utilizado na construção do sistema (isto é, o Elipse SCADA) incorpora a tecnologia OPC como servidor e como cliente. (c) O padrão OPC está se tornando rapidamente o padrão de comunicação adotado pelo mercado de automação industrial e pela indústria, e muitas soluções de automação que dependem das informações de chão-de-fábrica já utilizam OPC como condição inicial para comunicação de dados, segundo afirma Fonseca (2002). Isto confirma a relevância e importância desta tecnologia para a automação e para a indústria.

O padrão OPC é composto por um conjunto de especificações para comunicação no ambiente industrial, criado com a colaboração de vários dos mais importantes fornecedores de *hardware* e *software* para automação, em conjunto com a Microsoft. O padrão permite que aplicações de *software* troquem dados entre si de forma aberta e simplificada, assim como aplicações gerenciais e dados de chão-de-fábrica (OPC Foundation, 1998).

A OPC Foundation é a organização que gerencia o padrão OPC e conta com mais de 300 membros, incluindo alguns dos maiores fornecedores mundiais de sistemas de controle e automação. A primeira versão do padrão OPC surgiu em 1996 e foi desenvolvida sobre uma plataforma Microsoft pelos precursores da OPC Foundation, a saber: Fisher-Rosemount, Rockwell Software, Opto 22, Intellution e Intuitive Technology. Para Gaidzinski (2003), o objetivo principal da fundação é desenvolver um padrão de comunicação aberto e flexível, que permita aos usuários escolher entre uma grande gama de soluções, além de reduzir consideravelmente os custos de desenvolvimento e manutenção dos fornecedores de *hardware* e *software*.

A arquitetura do padrão OPC foi originalmente baseada na tecnologia COM (*Component Object Model*) e DCOM (*Distributed Component Object Model*) da Microsoft, e definiu padrões de objetos, interfaces e métodos para uso em controle de processos e aplicações de automação da manufatura. O objetivo para criação do padrão é criar um meio comum para que aplicativos de diferentes camadas do processo possam trocar dados e interagir de forma mais fácil possível (OPC Foundation, 1998). A figura 2.9 mostra as diferentes camadas de informação contempladas na arquitetura OPC.

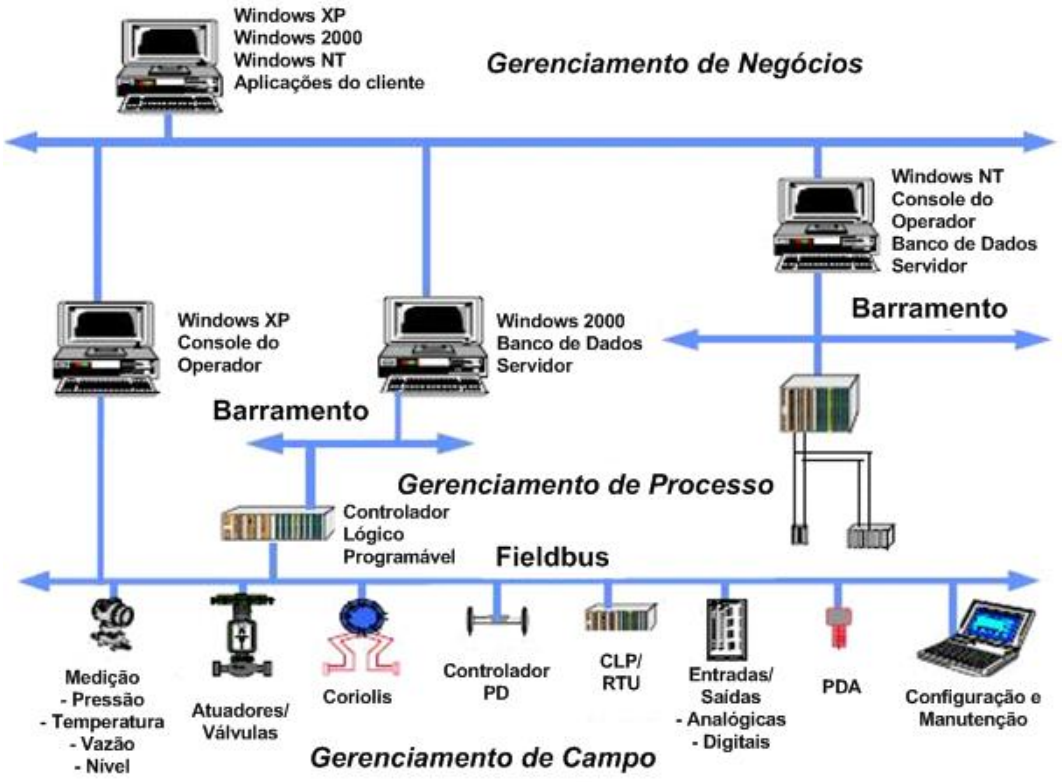


Figura 2.9. Camadas de informação englobadas pela tecnologia OPC.
(Fonte: Adaptado de OPC Foundation, 1998)

Segundo relata Fonseca (2002), a publicação das especificações para o padrão OPC, pela OPC Foundation, possibilitou o desenvolvimento de diversos produtos para automação industrial, que se beneficiaram das seguintes vantagens proporcionadas pelo padrão:

- Eliminação da necessidade de *drivers* de comunicação específicos (proprietários);
- Melhoria do desempenho da comunicação entre dispositivos de automação;
- Interoperabilidade entre sistemas de diversos fabricantes;
- Integração com sistemas MES (*Manufacturing Execution Systems*) e ERP (*Enterprise Resource Planning*);
- Padronização das interfaces de comunicação entre os servidores e clientes de dados em tempo real, facilitando a integração e manutenção dos sistemas;
- Redução dos custos e tempos para desenvolvimento de interfaces e *drivers* de comunicação, com conseqüente redução do custo de integração de sistemas;
- Facilidade de desenvolvimento e manutenção de sistemas e produtos para comunicação em tempo real.

2.5 SISTEMAS SCADA COM OPC

A arquitetura típica de um cenário de automação industrial, baseada em controle de processo, divide-se em três níveis de funções: supervisão, gerência de processos e gerência de dispositivos. A figura 2.10 apresenta as relações entre estes níveis de funções (coluna da esquerda) e as tecnologias de um sistema de supervisão e controle padrão (coluna da direita).

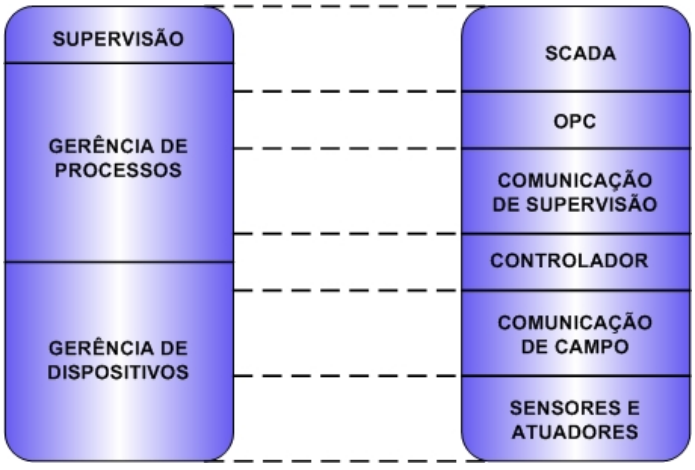


Figura 2.10. Arquitetura de um sistema de gerenciamento de processos industriais.
(Fonte: adaptado de PIRES et al, 2005)

Uma análise inicial da arquitetura do sistema de gerenciamento apresentado mostra que o padrão OPC (coluna da direita) é o responsável por estabelecer a interface entre o sistema SCADA e o módulo controlador. A tecnologia OLE (*Object Linking Embedding*) permite que uma aplicação crie informações e as apresente através de uma segunda aplicação (PIRES et al, 2005). O objeto criado na aplicação original pode ser conectado à segunda aplicação, que se referencia à aplicação original por meio de um ponteiro. O objeto também pode ser embarcado na segunda aplicação, e neste caso ele é copiado da aplicação original para a secundária. Na prática, o módulo OPC funciona como uma API (*Application Programming Interface*) suportando a comunicação entre o sistema SCADA e o módulo controlador, que pode prescindir dos controladores de diferentes fabricantes, aumentando a interoperabilidade do sistema. Os fabricantes de controladores fornecem seus próprios *drivers* OPC para serem incorporados ao servidor OPC do supervisório. Geralmente os módulos OPC são implementados com os protocolos da pilha TCP/IP (*Transmission Control Protocol/Internet Protocol*). A figura 2.11 descreve a arquitetura entre uma aplicação cliente e servidores OPC de diferentes fabricantes.

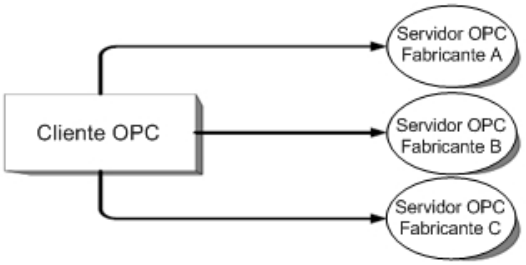


Figura 2.11. Arquitetura entre Cliente OPC e diversos Servidores OPC.
(Fonte: Adaptado de OPC Foundation, 1998)

As conexões entre diferentes sistemas SCADA normalmente são baseadas em uma rede local *Ethernet* e são controladas pelo módulo de Comunicação de Supervisão. A conexão entre o Controlador de Processo e os sensores, transdutores e atuadores em campo, é realizada pelo módulo de Comunicação de Campo. Em aplicações de automação e controle industriais típicas, baseadas em CLPs, o módulo de Comunicação de Campo utiliza comunicação ponto-a-ponto, como por exemplo, o protocolo ModBus (SEGURA, 2005).

O módulo de Supervisão SCADA é o responsável pelas funções de Supervisão e de Gerência de Processos. O módulo OPC, assim como os módulos de Comunicação de Supervisão e Controlador, são os encarregados da função de Gerência de Processos. O módulo Controlador também detém a responsabilidade compartilhada na função de Gerência de Dispositivos, em conjunto com o módulo de Comunicação de Campo e os elementos terminais (sensores, transdutores e atuadores).

As interfaces OPC podem ser utilizadas de diversas formas dentro de uma aplicação de supervisão e controle, embora inicialmente tenham sido concebidas para acessar dados de uma aplicação servidora. A figura 2.12 mostra um exemplo dessa diversidade de relacionamentos entre servidores e clientes OPC. No campo, dados podem ser adquiridos de sensores, transdutores e atuadores para um sistema SCADA, que ao mesmo tempo se comunica com a aplicação, também através do padrão OPC. Diversos aplicativos, ou sistemas SCADA, podem trocar dados dentro de uma rede, mesmo envolvendo diferentes fornecedores.

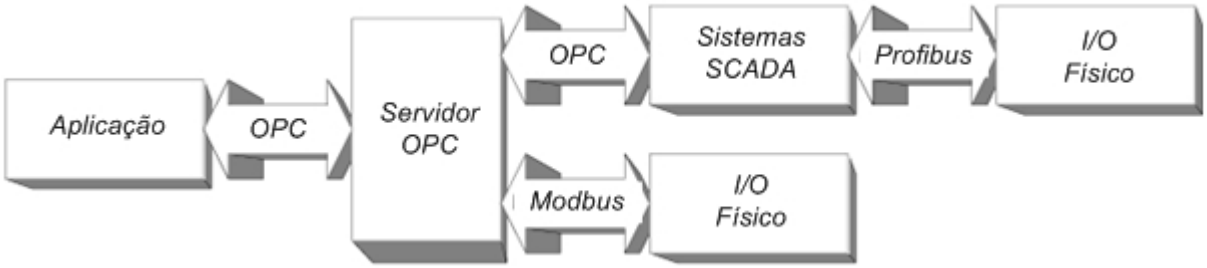


Figura 2.12. Exemplo de relacionamento entre Servidores e Clientes OPC.
(Fonte: Adaptado de OPC Foundation, 1998)

2.6 *SOFTWARE* ELIPSE SCADA

O *software* Elipse SCADA é uma ferramenta para o desenvolvimento de sistemas de supervisão e controle, que permite a criação e execução de aplicativos IHM e SCADA para uma grande quantidade de processos. Através da coleta de informações de qualquer tipo de equipamento de controle, os operadores podem monitorar e controlar todos os processos de chão-de-fábrica, bem como máquinas e recursos, gerenciando de forma interativa e transparente toda a produção. Os dados são apresentados de forma gráfica em tempo real, permitindo o tratamento das informações de forma simples e organizada. O Elipse SCADA pode trocar dados com vários equipamentos de aquisição de dados, como CLPs e RTUs, e com outros sistemas SCADA, através da comunicação OPC. Ela possui ainda a ferramenta ODBC, para troca de dados com bancos de dados relacionais.

Comercialmente o Elipse SCADA é oferecido em quatro versões (http://www.elipse.com.br):

- Elipse View: é a versão utilizada para a construção de interfaces de operação para monitoração e acionamento de equipamentos. Possui visualização de variáveis de forma gráfica, programação de *setpoints*, controle de acesso, visualização de alarmes, *scripts*,

servidor e cliente DDE (*Dynamic Data Exchange*). Esta é a versão mais simples do Elipse SCADA;

- Elipse MMI: além das funções oferecidas pela versão View, possui banco de dados proprietário, relatórios formatados, históricos, receitas, controle de alarmes e Controle Estatístico do Processo (CEP). É a versão indicada para qualquer porte de sistema, onde não haja necessidade de conexão com bancos de dados externos via ODBC ou *Data Access Objects* (DAO), ou quando não seja necessário enxergar outras estações através da rede;
- Elipse PRO: além das funções disponibilizadas na versão MMI, permite trocar dados em tempo real com outras estações através do servidor Elipse TCP/IP, conectar-se com bancos de dados, realizar comandos e programar *setpoints* através de rede local ou linha discada. Permite a comunicação com equipamentos e sistemas via OPC e conexão com *softwares* de controle SoftPLC de terceiros;
- Elipse Power: esta versão foi desenvolvida para aplicação em sistemas de geração, transmissão e distribuição de energia. Possui recursos avançados como a conexão com IDEs e RTUs através de protocolos como IEC 870-5 e DNP 3.0. O Elipse Power permite o sequenciamento de eventos com precisão de 1 ms (um mili segundo), oscilografia e telesupervisão. É possível também sincronizar o relógio do computador que controla o processo com equipamentos remotos via GPS.

São três os módulos de operação do Elipse SCADA: *Master*, *Runtime* e Configurador. O módulo *Runtime*, ou módulo ativo, permite apenas a execução dos aplicativos. Os módulos *Master* e Configurador permitem a criação, desenvolvimento e teste de aplicativos. O *software* supervisório pode ser baixado gratuitamente na página da Elipse na Internet (http://www.elipse.com.br/elipse/downloads.aspx?id_produto=2&titulo_secao=Elipse%20SCADA). A versão utilizada para o desenvolvimento dos gerenciadores nesta dissertação é a v2.28 build 024.

2.6.1. Aplicativos no Elipse SCADA

Os aplicativos desenvolvidos no Elipse SCADA utilizam uma série de objetos de tela (*slider*, *button*, *setpoint*, *gauge*, *text*, *display*, *bar*, *animation*, *browser*, etc). Cada objeto possui um conjunto de propriedades que pode ser configurado pelo usuário. A figura 2.13 mostra o conjunto de objetos na barra de tarefas do supervisório e as propriedades do objeto botão.

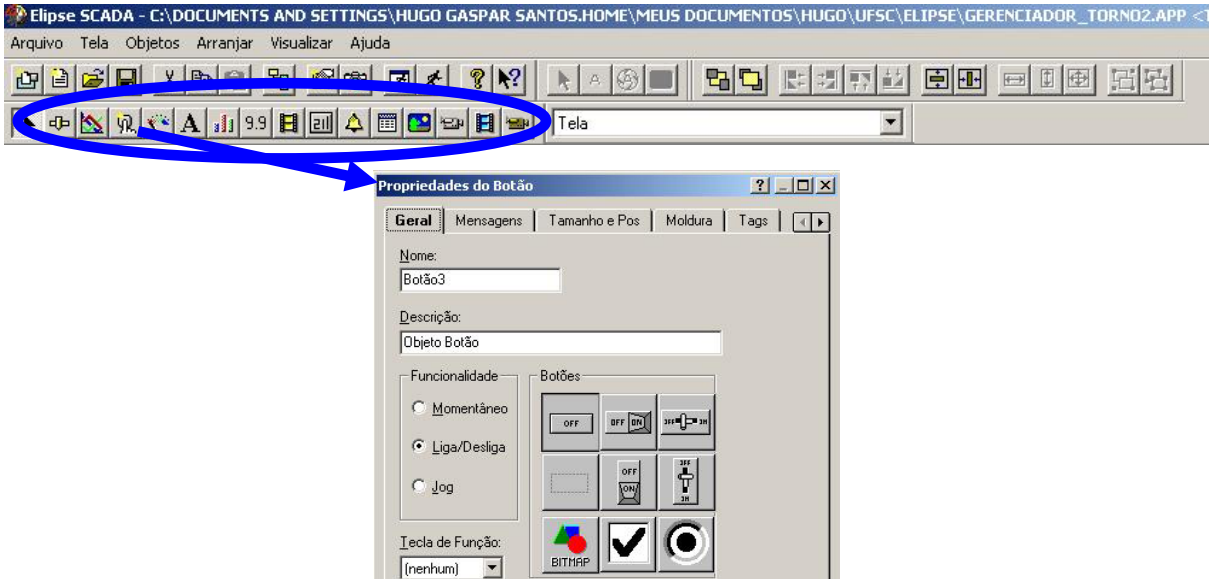


Figura 2.13. Barra de tarefas e propriedades do objeto botão.

As variáveis do processo que se deseja monitorar ou controlar recebem a denominação de *tags*. Os objetos de tela se associam às *tags* para oferecer uma interface ao usuário, de tal forma que este possa interagir com o processo monitorado. Através do *organizer*, que é uma janela do Elipse SCADA que permite visualizar toda a estrutura do aplicativo, é possível visualizar as *tags* utilizadas, os objetos de tela, os *drivers* configurados, os servidores OPC, etc. A figura 2.14 mostra o *organizer* de um aplicativo e os diferentes tipos de *tags* oferecidas pelo supervisório.

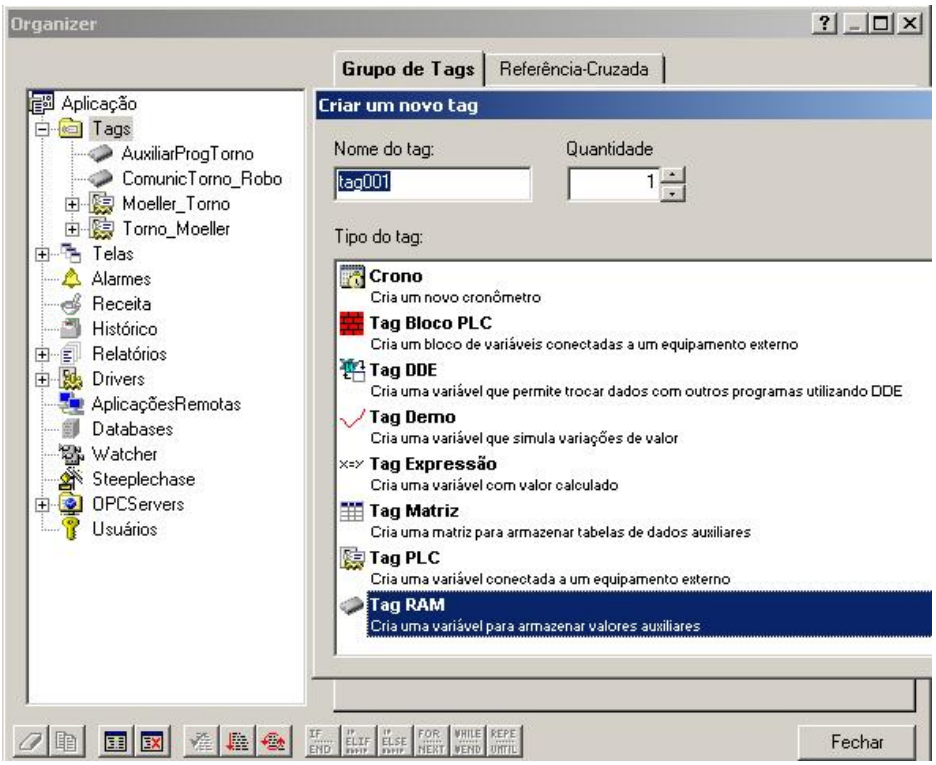


Figura 2.14. *Organizer* e tipos de *tags* do Elipse SCADA.

O supervisório permite a comunicação com equipamentos externos, como CLPs ou RTUs. Também é possível trocar dados com outras aplicações SCADA através de uma rede de comunicação. Em ambos os casos, é necessário atribuir e configurar um *driver* (fornecido pela Elipse) que permita a troca de dados de forma correta entre o aplicativo e o equipamento externo. A figura 2.15 mostra a configuração de alguns *drivers* utilizados em um aplicativo.

Figura 2.15. *Drivers* e configuração no *organizer* do Elipse SCADA.

2.6.2. OPC no Elipse SCADA

O Elipse SCADA pode atuar como cliente OPC, e também permite que a base de dados criada em servidores OPC seja importada para uma base de dados interna. Várias conexões podem ser feitas com diferentes servidores OPC. Segundo Maciel (2005), a tecnologia OPC implementa um mecanismo que provê dados de algum dispositivo para uma base de dados configurada em um servidor OPC, e permite que qualquer aplicação cliente tenha acesso à mesma base de dados.

A figura 2.15 mostra na árvore do *organizer* o item OPCServers. Ao selecionar esta opção, abre-se uma segunda tela que permite procurar e configurar os servidores OPCs presentes na rede de

comunicação. A figura 2.16 apresenta um exemplo de servidores OPCs presentes em uma rede de comunicação e a configuração do servidor selecionado.

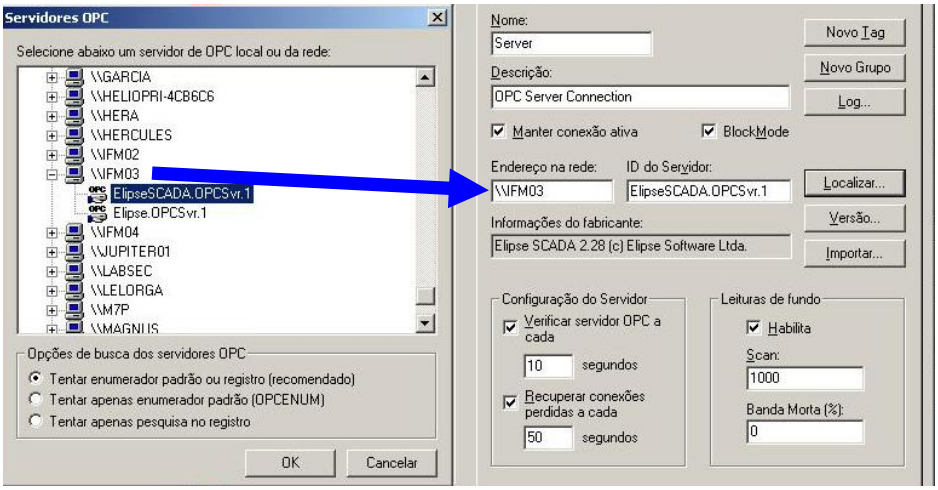


Figura 2.16. Lista de Servidores OPC e configuração.

Após informar o endereço do servidor OPC na rede e configurar parâmetros de comunicação, as *tags* da aplicação servidora podem ser importadas (figura 2.17).

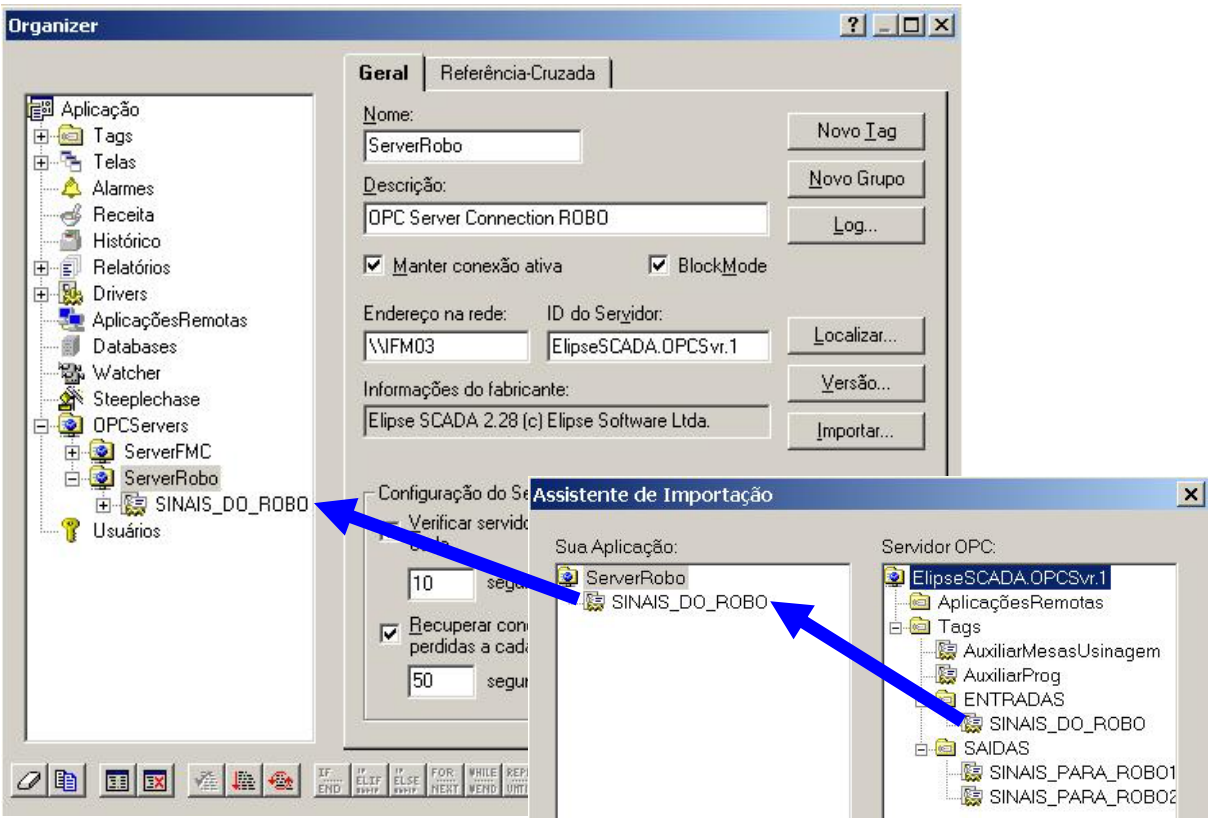


Figura 2.17. Importação de uma *tag* do servidor OPC.

Se o sistema operacional utilizado no servidor, ou no cliente OPC, for o Windows XP SP2 (Service Pack 2), será necessário alterar algumas configurações originais do Windows para permitir a correta troca de dados via OPC. Hiller (2004) relata com detalhes todas as modificações necessárias para habilitar a comunicação entre aplicações OPC no *software* Elipse SCADA. De forma resumida estas alterações se referem a:

- Desativação do *Firewall* do Windows;
- Se não for possível desativar o *Firewall,* então é necessário incluir na lista de exceções do *Firewall* as portas DCOM-TCP-135 e DCOM-UDP-135, além dos programas Elipse32.exe e ED_OPC.exe.
- Reconfiguração do DCOM através do seguinte roteiro: abrir o gerenciador de "Serviços de componente" (dcomcnfg) – Computadores – Meu computador (Propriedades) – Segurança COM (figura 2.18). Neste ponto é necessário editar os novos limites de "Permissões de acesso" (habilitar acesso remoto) e "Permissões de Inicialização e Ativação" (habilitar inicialização remota e ativação remota).

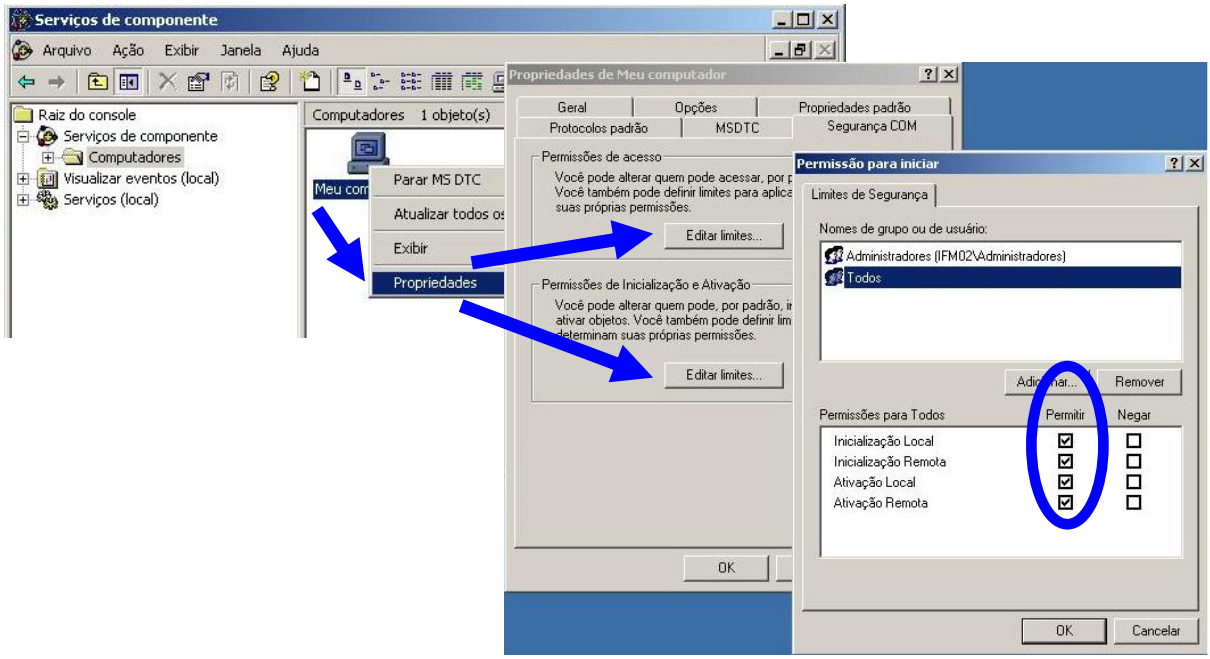


Figura 2.18. Configuração do DCOM.

3. DESCRIÇÃO GERAL DO TRABALHO

Este capítulo apresenta o laboratório onde foi realizado o projeto de construção da FMC, seus principais recursos produtivos e as condições de contorno. Os objetivos gerais e específicos do projeto são aprofundados, além das motivações que levaram a escolha do tema e a relevância do assunto, tanto no âmbito acadêmico quanto industrial.

O uso de uma arquitetura aberta e modular para a integração da célula, baseada no uso de gerenciadores desenvolvidos a partir de sistemas SCADA, o uso de CLPs para integração dos recursos produtivos e a comunicação entre os gerenciadores utilizando o padrão OPC, são algumas das propostas apresentadas e discutidas.

3.1 O LABORATÓRIO

O laboratório utilizado para o desenvolvimento desta dissertação se encontra nas dependências da SOCIESC em Joinville, SC. A figura 3.1 apresenta uma vista parcial do laboratório.

Figura 3.1. Vista parcial do laboratório.

O laboratório conta com equipamentos suficientes para a configuração de uma Célula Flexível de Manufatura, porém antes do início do projeto de pesquisa não havia nenhum tipo de integração que permitisse movimentar a matéria-prima e produzir peças nos CNCs de forma autônoma. Nos últimos

anos diversos trabalhos de pesquisa, e outros de conclusão de curso de graduação, foram realizados no sentido de reunir informações e realizar melhorias no laboratório, sempre direcionados a uma futura integração da célula. Este trabalho de pesquisa, em relação aos procedimentos técnicos adotados, é considerado também como uma pesquisa-ação, justamente porque foi concebido como resolução de um problema coletivo (a integração dos equipamentos do laboratório da SOCIESC para formação de uma FMC) e foi realizado com a participação cooperativa de outros trabalhos de pesquisa realizados no mesmo laboratório.

3.2 DISTRIBUIÇÃO DOS EQUIPAMENTOS NO LABORATÓRIO

A disposição física dos equipamentos no laboratório não permite mudanças significativas no *layout*. O armazém automático ocupa a maior parte do espaço físico. O torno e o centro de usinagem CNC foram agrupados próximos às mesas dos berços de usinagem do armazém, e o robô industrial ABB foi fixado eqüidistante das portas das máquinas CNC. O espaço de trabalho do robô alcança as mesas dos berços de usinagem do armazém e o interior das máquinas CNC. A figura 3.1 permite visualizar a posição do robô, dos CNCs e os berços de usinagem do armazém automático.

Os recursos computacionais do laboratório são compostos por dois CLPs e seis computadores, todos interligados por meio de uma rede *Ethernet*.

3.2.1 Armazém Automático

O armazém automático é composto por uma estrutura porta-*pallet*, esteiras de entrada, esteiras de saída/rejeito, esteiras de usinagem e transelevador (figura 3.2). A função do armazém é receber a matéria-prima, acondicionada em *pallets*, e armazená-la de forma automática na estrutura porta-*pallet*. No processo de entrada de material, o operador identifica a matéria-prima com um código previamente cadastrado e o número do *pallet* (identificado em código de barras). O operador também pode solicitar a saída do material armazenado, bastando para isso informar o código do material desejado. Para o processo de usinagem, o operador digita o código do produto final (já processado). O sistema verifica qual é a matéria-prima necessária para a fabricação do produto e a encaminha às mesas dos berços de usinagem para processamento nas máquinas CNC.

A movimentação entre as esteiras e a estrutura porta-*pallet* é realizada pelo transelevador, um robô cartesiano que se movimenta sobre um trilho ao longo de todo o armazém. Atualmente os *pallets* têm a capacidade de acomodar somente uma peça. O armazém comporta um total de 48 *pallets*.

A interface entre o operador e o armazém é realizada por um aplicativo desenvolvido pelo fabricante do equipamento no *software* Elipse SCADA.

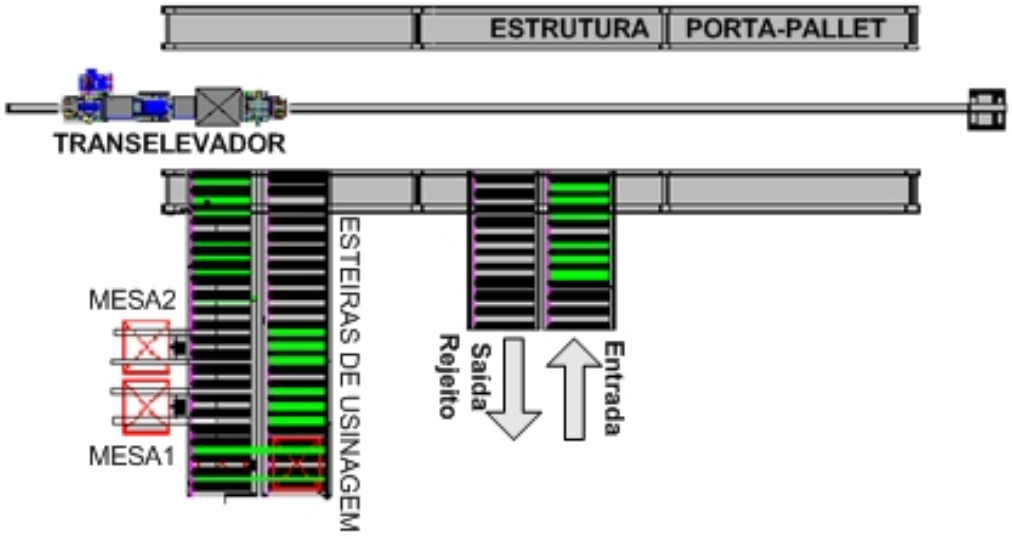


Figura 3.2. Armazém automático

3.2.2 Robô ABB

O robô industrial ABB, modelo IRB 2400/10 é o responsável por movimentar as cargas desde os berços de usinagem do armazém até as máquinas CNC. Após o processamento da matéria-prima, o robô deve retirar as peças do interior das máquinas CNC e reconduzi-las aos berços de usinagem do armazém. Para carregar as peças, o robô conta com uma garra de acionamento pneumático, que foi desenvolvida especificamente para esta função.

O conjunto do robô é composto por duas partes: o controlador e o manipulador móvel. Os programas do robô podem ser inseridos no controlador por meio de um disquete, ou pela unidade de programação (*Teach Pendant*). O manipulador móvel apresenta 6 graus de liberdade e capacidade de carga total de 10 kg. A figura 3.3 apresenta o controlador do robô e a unidade de programação.

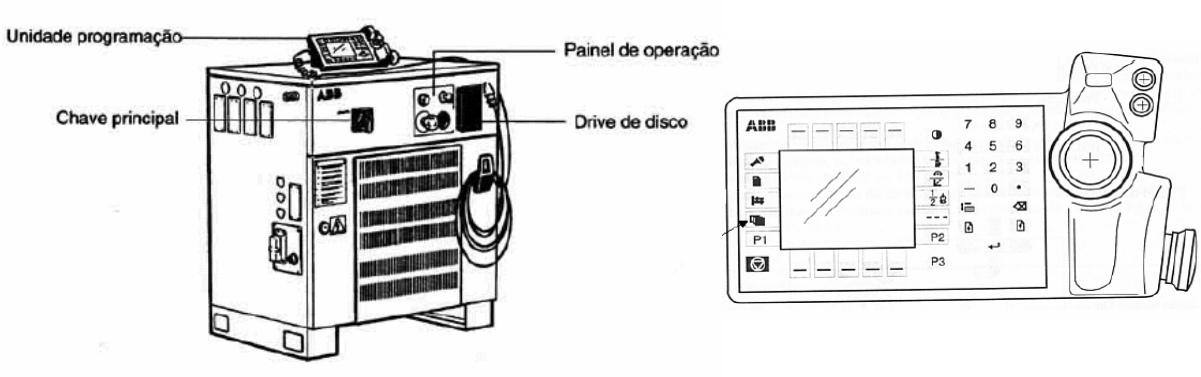


Figura 3.3. Controlador e unidade de programação do robô.

3.2.3 Equipamentos CNC

Os equipamentos CNC são os responsáveis pela transformação da matéria-prima em produtos intermediários ou acabados. O laboratório conta com um torno CNC, marca Feeler, modelo FTC10 e um centro de usinagem também Feeler, modelo FV600. O estudo detalhado do espaço físico restante no laboratório indica a possibilidade de instalar um terceiro equipamento CNC. O espaço de trabalho do robô alcança esta hipotética terceira máquina, razão pela qual ela será incluída na modelagem da célula (com o nome de CNC R1), apesar de não estar fisicamente presente.

O torno possui um trocador de ferramentas tipo revólver, com capacidade para até 8 ferramentas. Já o centro de usinagem, de três eixos, está equipado com um sistema de troca de ferramentas tipo carrossel. A figura 3.4 apresenta o torno e o centro de usinagem.

Figura 3.4. Torno e centro de usinagem CNC.

3.3 O DESAFIO DA INTEGRAÇÃO

O maior problema para a formação de uma Célula Flexível de Manufatura, sem dúvida, está na integração dos diferentes equipamentos. Individualmente estas máquinas apresentam poucos recursos de comunicação com o meio externo, exceção feita ao armazém automático. O robô, as máquinas CNC e o armazém automático possuem *hardwares* diferentes e nenhuma padronização de comunicação. Ressalte-se o fato de que os equipamentos do laboratório são todos semelhantes aos encontrados na indústria, sem nenhum dispositivo extra, ou seja, não foram preparados para a integração e necessitam ser modificados para permitir um mínimo de interatividade entre eles. Na literatura pesquisada, há vários exemplos de formação de Células Flexíveis de Manufatura, a maioria com finalidade didática: a dissertação de Guilherme Vieira que trata da construção de uma FMC didática (VIEIRA, 1996); a construção da FMC do laboratório GRACO, da UnB (TEIXEIRA, 2006); a modelagem e implementação de um sistema de supervisão na FMC da Universidade Técnica de Lisboa (RIBEIRO e ELVAS, 2004); a implantação de um laboratório de Automação da Manufatura no CEFET-PR (BOARETTO et al, 2004); os FMSs na Yamazaki Mazak (SLACK et al, 2002). Alguns relatos contendo descrições sumárias e fotos de FMCs podem ser encontrados na Internet, entre os quais:

- ✓ O laboratório FMS da Pontifícia Universidad Javeriana de Cali, na Colômbia (http://atlas.puj.edu.co/ftp/centros/cap/manual_fms.pdf.);
- ✓ Os laboratórios FMS do SENAI (http://www.rs.senai.br/mecatronica/instalacoes.htm, http://www.sp.senai.br/mecatronica/Index.Asp?LinkPage=EscolaLabs, http://www.joinville.senai.br/pags/laboratorios.php?lab= robô);
- ✓ O laboratório da Engenharia da Produção, na PUC-PR (http://www.produtronica.pucpr.br/Intranet/ laboratorios/lista-labs.php#LAS01);
- ✓ O laboratório FMS-200 da Universidad de Oviedo, na Espanha (http://www.isa.uniovi.es/genia/spanish/publicaciones/celula.pdf);
- ✓ O laboratório CIM da Pennsylvania State University (http://www.engr.psu.edu/cim/FAME/CIMLAB/cim_home.html).

De forma geral, as referências encontradas a respeito de manufatura flexível podem ser enquadradas em três grupos bem definidos:

- A modelagem, simulação e controle de Sistemas Flexíveis de Manufatura (MUŠIČ e MATKO, 1998), (ZHOU e VENKATESH, 2000), (ROHDE e BORENSTEIN, 2004), (GÓMEZ e LORENA, 2006), (SAVARIS e POSSAMAI, 2005), (QUEIROZ e CURY, 2002), (TORRICO e CURY, 2004), (PINA et al, 2006). Estes trabalhos apresentam um cunho eminentemente teórico a respeito de modelagem, simulação e controle de Sistemas Flexíveis de Manufatura, sem implementações de ordem prática.

- Implantações de Células e Sistemas Flexíveis de Manufatura com finalidade didática. Há diversos relatos de trabalhos de implantação de FMSs e FMCs em laboratórios ligados a instituições de ensino. A grande maioria deles apenas descreve as potencialidades do sistema, sem aprofundar detalhes sobre seu funcionamento, integração, interfaces ou métodos de controle adotados. Em alguns casos, apenas fotos dos laboratórios são mostradas.

- Implantação de FMSs comerciais. Existem informações sobre a utilização de FMSs em diversas empresas, como: Yamazaki Mazak (SLACK et al, 2002), Detroit Diesel Allison, Vought Aircraft (FERREIRA, 1998), General Electric, GM, Hughes Aircraft, Allen Bradley, Chrysler, Pratt and Whitney's (ZHOU e VENKATESH, 2000). Alguns dos principais fabricantes de FMS são: Yamazaki Mazak (http://www.mazak.jp), Kearney & Trecker Company (http://www.equipmentmls.com), Heller (http://www.heller-machinetools.com) e SMC Internacional Training (http://www.smceu.com).

Muitos e meritórios trabalhos foram publicados a respeito do quê são as Células e os Sistemas Flexíveis de Manufatura; sobre modelagem, simulação e controle desses sistemas; e sobre as dimensões da flexibilidade (de produto, sequenciamento, volume, processo, produção, etc.). Em contrapartida, poucos são os trabalhos encontrados que descrevem como integrar efetivamente os diversos recursos produtivos para a formação de uma Célula, ou Sistema Flexível de Manufatura. Uma exceção a esta regra é o trabalho realizado por Evandro Teixeira, na modelagem e integração de um robô, um centro de torneamento CNC e um micrômetro laser, para a formação de uma FMC, na UnB (TEIXEIRA, 2006). Os FMSs e FMCs comerciais normalmente são construídos sobre plataformas proprietárias, ou seja, adotam um padrão de *hardware* e *software* onde o domínio da tecnologia não é transferido ao usuário. Toda modificação na estrutura, ou no processo, demanda a intervenção do fornecedor do equipamento.

O projeto de pesquisa para integração da Célula Flexível de Manufatura da SOCIESC parte da premissa de construir um sistema aberto, em que os diferentes recursos produtivos sejam reduzidos a módulos, representados por um conjunto mínimo de interfaces. Gerenciadores serão construídos para comandar as ações dos módulos, e trocarão dados entre si, por meio de um padrão de comunicação industrial aberto e que não exija o uso de protocolos proprietários. Como contribuição científica resultante da integração dos equipamentos, apresenta-se uma diretriz, composta por um conjunto de instruções e procedimentos, para formação de uma FMC. Neste caso, a pesquisa aplicada tem como objetivo gerar conhecimentos para aplicação prática e é dirigido à solução do problema específico de como integrar de forma física e lógica os equipamentos do laboratório da SOCIESC, para formação de uma FMC. Mantendo a devida coerência com os objetivos inicialmente propostos, esta dissertação apresenta em detalhes todos os aspectos técnicos envolvidos na construção do sistema supervisório, permitindo utilizá-lo não somente como uma metodologia de propósito geral, mas também como um roteiro para implementações de ordem prática.

Para a construção dos gerenciadores, um para cada equipamento presente na célula, será utilizado um *software* de supervisão e controle (Elipse SCADA). Controladores lógicos programáveis (CLPs) farão a ligação entre os gerenciadores e os CNCs, que possuem somente uma porta serial para aquisição de programas de usinagem. O robô, que está equipado com um cartão de comunicação padrão Profibus, utilizará este recurso para a comunicação com um CLP, que por sua vez, estará conectado ao gerenciador. O armazém automático já possui uma interface com o usuário (IHM), que será modificada para cumprir o papel de gerenciador. A figura 3.5 apresenta um resumo da arquitetura proposta para integração da célula.

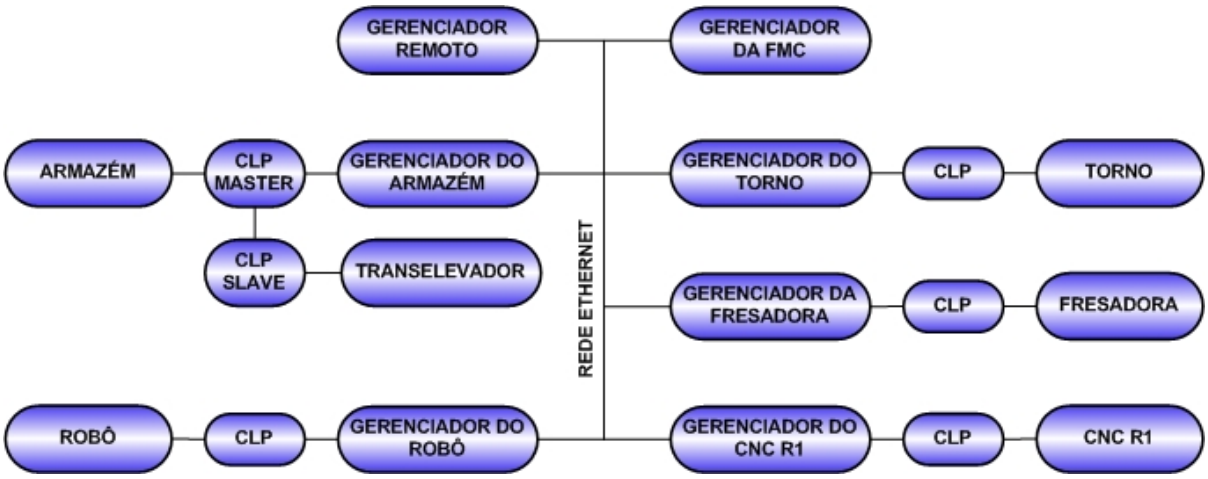


Figura 3.5. Arquitetura da proposta de integração da célula.

Cada equipamento da célula necessita de adaptações em maior ou menor grau, visando a integração. O torno e o centro de usinagem, por exemplo, possuem portas que são operadas manualmente. Para que estas máquinas possam receber a matéria-prima carregada pelo robô, as portas deverão ser automatizadas. Os equipamentos CNC foram construídos originalmente para funcionar com a ajuda de um operador. Modificações de *hardware* e de *software* devem ser realizadas para eliminar, ou minimizar, a necessidade de intervenção deste operador, conferindo um grau de autonomia operacional aos equipamentos. Pelini (2001) relata uma adaptação semelhante, realizada em um torno CNC, para a implantação de um sistema integrado de manufatura, na Universidade de Caxias do Sul.

3.4 ROTEIRO DE TRABALHO

O primeiro passo para a integração da célula é determinar que tipo de controle será adotado. Segundo Pels et al. (1997), a evolução dos sistemas de controle se caracteriza por um marcante aumento da autonomia dos módulos de controle, pelo enfraquecimento das relações mestre-escravo e pela redução do fluxo de informação entre os módulos controladores. A figura 3.6 apresenta um resumo da evolução histórica das arquiteturas dos sistemas de controle.

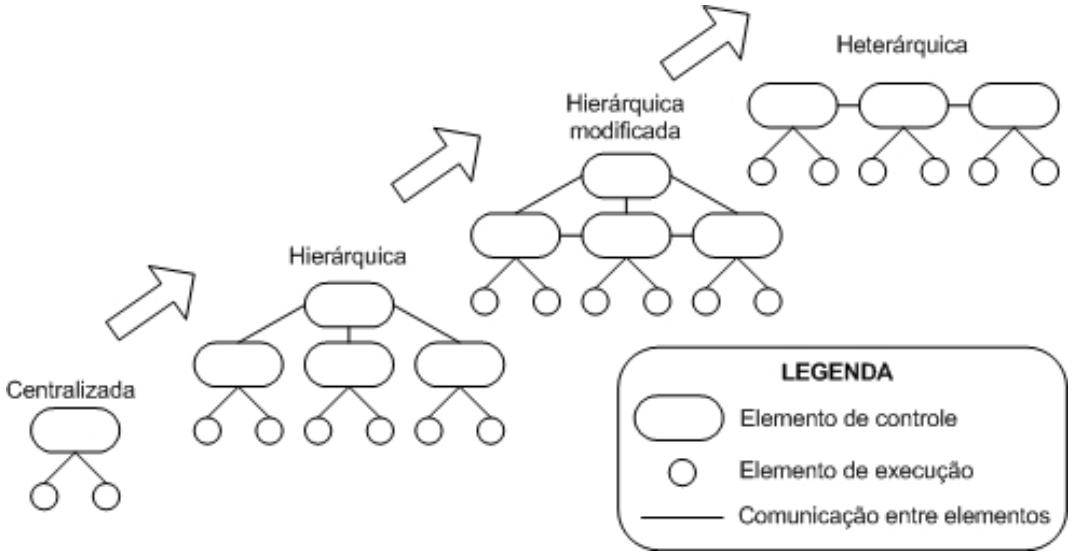


Figura 3.6. Evolução histórica das arquiteturas dos sistemas de controle
(Adaptado de Pels et al, 1997)

A arquitetura centralizada é uma das primeiras arquiteturas utilizadas no projeto de sistemas de controle. O processamento é realizado por um único módulo de controle que também centraliza

todo o fluxo de informações. As principais vantagens desta arquitetura são o gerenciamento e controle otimizados.

A evolução natural da arquitetura centralizada para a arquitetura hierárquica, dispôs os elementos de controle em uma estrutura piramidal e multicamadas, desta forma o processamento deixa de ser centralizado e passa a ser distribuído. Na arquitetura hierárquica estabelece-se uma relação mestre-escravo entre as diferentes camadas de controle, ou seja, o processo decisório inicia-se de cima para baixo (*top-down*) e os resultados decorrentes das ações são repassados de baixo para cima (*bottom-up*). Esta arquitetura de controle facilita o desenvolvimento modular, mas também é susceptível a perturbações. A arquitetura hierárquica modificada mantém todas as características da arquitetura hierárquica e acrescenta interações entre os módulos que estão dispostos em um mesmo nível hierárquico. Este arranjo permite diminuir as perturbações, pois reduz o tráfego de informações no controlador principal.

Na arquitetura heterárquica, a relação mestre-escravo estabelecida entre os módulos controladores é substituída por uma relação cooperativa, onde todos os elementos de controle atuam no mesmo nível e a tomada de decisão é local e autônoma. Segundo Friedrich (1996), este tipo de controle pressupõe a não existência de relacionamentos do tipo mestre-escravo, ou seja, o sistema é composto por entidades inteligentes que cooperam para realizar seus objetivos. A descentralização do processo decisório permite uma maior autonomia e desempenho do sistema, apesar de dificultar a análise geral do sistema, pela falta de uma relação hierárquica entre os módulos de controle (TEIXEIRA, 2006).

A arquitetura escolhida para o sistema de controle utilizado para a construção da Célula Flexível de Manufatura foi a arquitetura heterárquica, onde todos os gerenciadores se encontram no mesmo nível, com autonomia de decisão e agindo de forma cooperativa. Estas características são desejáveis, pois cada módulo deve possuir o máximo de autonomia possível, de maneira a efetuar localmente o maior número de ações necessárias para a integração da célula.

O passo seguinte à escolha da arquitetura do sistema de controle é definir um conjunto de interfaces para cada gerenciador. A filosofia de funcionamento dos gerenciadores prevê o máximo de autonomia possível para cada um deles. O robô, por exemplo, deverá estar com todos os programas de movimentação previamente carregados. O gerenciador apenas indicará qual programa deve ser executado. A mesma regra vale para os CNCs. Desta forma é possível reduzir o número de mensagens circulando entre os diversos gerenciadores. Para definir um número mínimo de interfaces para os gerenciadores, serão decompostas as ações executadas por cada equipamento e identificados os

sincronismos necessários entre as ações. A interdependência entre elas e o grau de autonomia de cada gerenciador é que determina o conjunto de mensagens necessárias para a integração.

Com a definição do tipo de controle a ser utilizado, e das interfaces para cada gerenciador, a etapa seguinte é efetuar a modelagem da célula. A modelagem a eventos discretos, utilizando Redes de Petri Interpretadas, fornece um modelo fiel do comportamento dinâmico e das relações de dependência entre os recursos, servindo de base para o desenvolvimento dos códigos do controle supervisório no *software* Elipse SCADA. As condições necessárias para o disparo de cada transição na modelagem são representadas por um conjunto de eventos no *software* Elipse SCADA, que, se satisfeitos, disparam a execução de um programa (*script*). As ações decorrentes do disparo da transição na modelagem são representadas pela execução do código do *script* no supervisório.

A integração da célula, baseada nos gerenciadores, será feita em duas etapas. A primeira delas, denominada integração horizontal, deve conectar cada gerenciador a seu respectivo equipamento. O robô e os equipamentos CNC necessitam de um CLP para trocar informações com os gerenciadores. O armazém automático já possui uma interface com o usuário, mas que precisa de alterações na estrutura do supervisório e no banco de dados. No caso do robô, por exemplo, programas de movimentação de peças para carga e descarga das máquinas CNC deverão ser escritos. O CLP que faz a ligação entre o gerenciador e o robô também deverá ser configurado e programado. A necessidade das máquinas CNC é semelhante ao robô, com o acréscimo de alterações nas características de *hardware*, tais como: automatização das portas, reprogramação do CLP interno, inclusão de válvulas, sensores, atuadores e banco de relés.

A segunda parte, denominada integração vertical, compreende a conexão de todos os gerenciadores numa rede de comunicação *Ethernet*. Nesta etapa os gerenciadores devem trocar informações entre si, utilizando o padrão de comunicação industrial OPC, para estabelecer os sincronismos necessários ao funcionamento da célula.

Dois novos gerenciadores serão desenvolvidos, além daqueles que comandam os recursos produtivos da célula, para fazerem o papel de interface com os usuários. O primeiro deles, o gerenciador FMC, deve fornecer ao usuário todas as informações sobre os estados dos equipamentos, dos pedidos efetuados, emitir relatórios, alarmes e permitir que o usuário insira seus pedidos de fabricação na FMC. O segundo, chamado de gerenciador remoto, tem a função de permitir que um usuário que se encontre fisicamente distante da célula, porém dentro da rede de comunicação, possa também inserir seus pedidos de fabricação. A figura 3.6 apresenta um resumo da proposta de integração para a FMC.

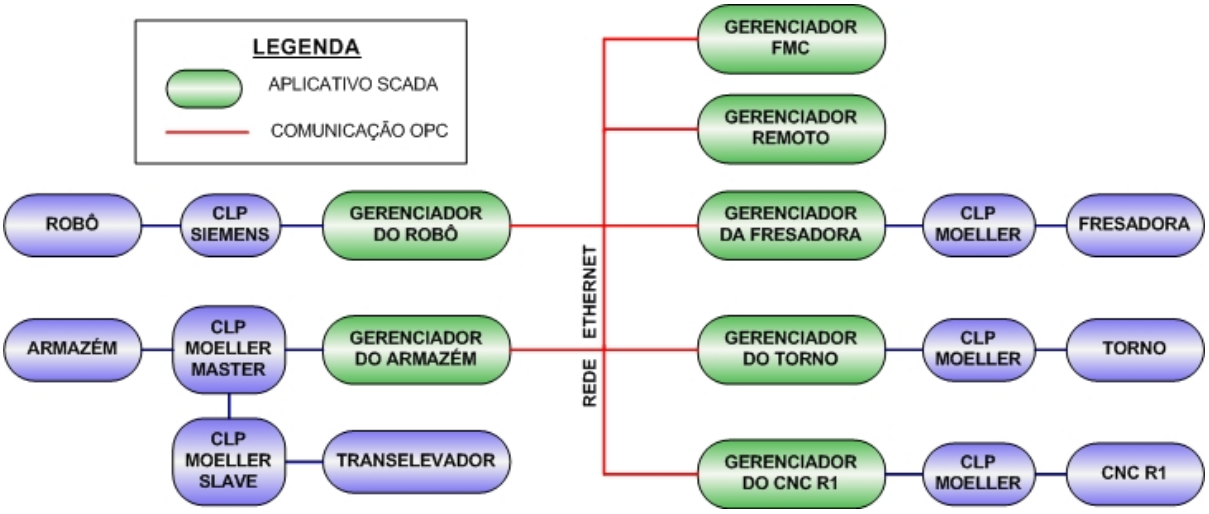


Figura 3.7. Arquitetura para integração da FMC

Para validar a integração da FMC, será simulada a fabricação das peças do jogo de xadrez (peão, torre, cavalo, bispo, dama e rei). As peças brancas serão fabricadas com um tipo de matéria-prima (alumínio – código MP001), enquanto as pretas utilizarão o cobre ou latão (código MP002), como matéria-prima para a fabricação. A transformação da matéria-prima nas peças de xadrez será realizada pelo torno e pelo centro de usinagem CNC. As duas mesas do berço de usinagem do armazém serão as responsáveis por fornecer os *pallets* contendo a matéria-prima para processamento nas máquinas CNC. A distribuição da fabricação das peças entre o torno e o centro de usinagem foi feita de forma a mostrar a flexibilidade do sistema, e não considerando a melhor estratégia real de fabricação para cada peça. A tabela 3.1 apresenta a relação entre os códigos de matéria-prima, peças, mesas de usinagem e máquinas CNC utilizadas para a fabricação.

Tabela 3.1. Relação entre peças, códigos, mesas e CNC.

Peça	Código	Matéria-prima	Mesa	CNC
Peão Branco	PBR	MP001	1	Torno
Peão Preto	PPR	MP002	1	Torno
Torre Branca	TBR	MP001	2	Fresadora
Torre Preta	TPR	MP002	2	Fresadora
Cavalo Branco	CBR	MP001	1	Fresadora
Cavalo Preto	CPR	MP002	1	Fresadora
Bispo Branco	BBR	MP001	2	Torno
Bispo Preto	BPR	MP002	2	Torno
Dama Branca	DBR	MP001	1	Torno
Dama Preta	DPR	MP002	1	Torno
Rei Branco	RBR	MP001	2	Fresadora
Rei Preto	RPR	MP002	2	Fresadora
M1R1	M1R1	MP001	1	CNC R1
M2R1	M2R1	MP002	2	CNC R1

O CNC R1 não está fisicamente presente na célula, porém a área onde ele hipoteticamente seria instalado está delimitada. O robô foi programado para transportar peças das mesas dos berços de usinagem do armazém, até o CNC R1, enquanto as peças já processadas são retornadas para o *pallet* correspondente. Dois códigos são os responsáveis por esta movimentação: (a) M1R1 executa o carregamento da máquina CNC R1, a partir da mesa1; e (b) M2R1: carregamento da mesma máquina a partir da mesa2.

4. MODELAGEM DA FMC

Este capítulo aborda os aspectos relacionados à definição do comportamento dinâmico da Célula Flexível de Manufatura (FMC), ou seja, a capacidade do sistema em atingir um determinado estado. Também são analisados os relacionamentos entre os diferentes recursos produtivos da célula, através do envio e recebimento de mensagens, os eventos gerados a partir dessas informações, os pré-requisitos e as ações associadas para a geração dos sincronismos necessários ao funcionamento da FMC. Em decorrência desta análise são definidas as interfaces mínimas de comunicação entre os diferentes recursos da célula. A partir da modelagem é possível visualizar o sistema em sua totalidade, especificar seu comportamento, definir os aspectos limitantes e documentar os estados e as decisões. A técnica utilizada para modelar a célula é a modelagem a eventos discretos, adotando o mecanismo formal de descrição das Redes de Petri Interpretadas (RdPI).

4.1 A CÉLULA FLEXÍVEL DE MANUFATURA

A disposição física dos equipamentos que compõem a FMC já estava definida antes do início do projeto de integração. Não havia possibilidade de mudanças significativas no *layout* devido a restrições de espaço. Apesar de o laboratório contar com os equipamentos necessários para a formação de uma Célula Flexível de Manufatura, não havia nenhum tipo de integração entre os equipamentos que possibilitasse uma ação autônoma e coordenada para a produção de peças. A figura 4.1 apresenta vistas parciais da célula, identificando seus principais componentes.

Os equipamentos de processamento de material são o torno CNC, o centro de usinagem CNC (fresadora) e um terceiro CNC virtual, denominado CNC R1. O robô é o responsável pelo manuseio e transporte de material entre o armazém automático e os equipamentos CNC. O sistema de armazenamento automático é o responsável por armazenar a matéria-prima e as peças acabadas e por fornecer a matéria-prima para ser processada nos equipamentos CNC. Todos os materiais são devidamente acondicionados em *pallets* no armazém.

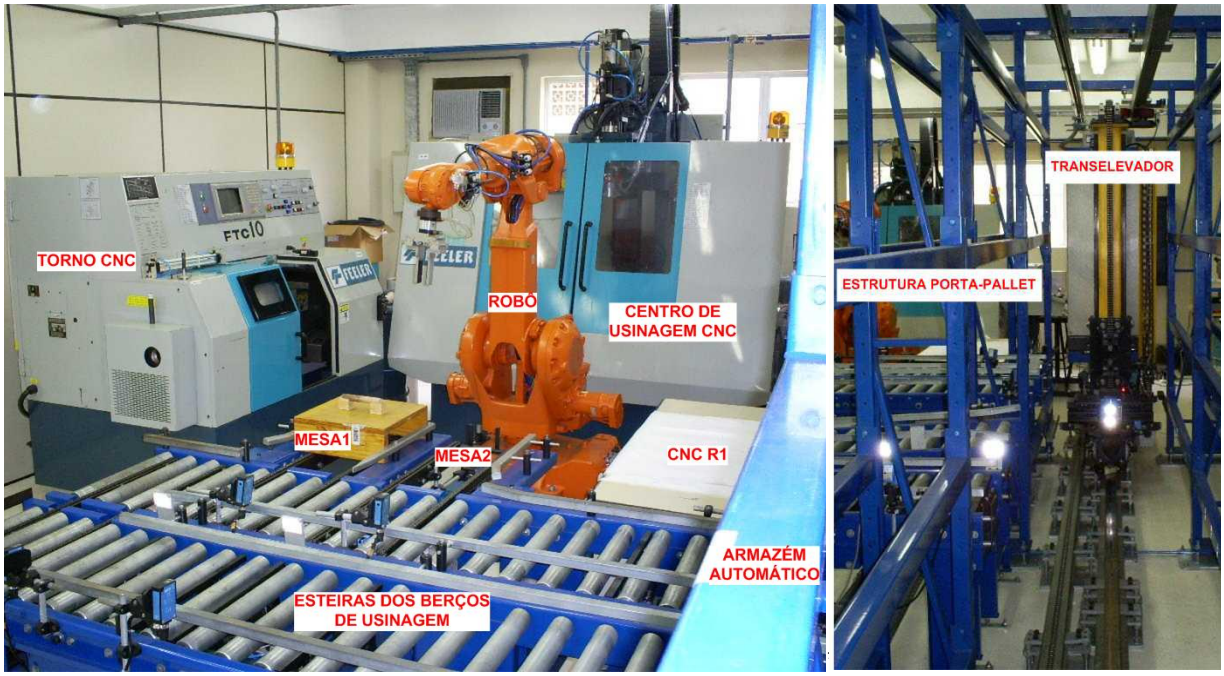


Figura 4.1. Identificação dos equipamentos da célula.

Na integração da célula, quando houver uma solicitação de fabricação, o armazém automático deve conduzir o *pallet* contendo a matéria-prima apropriada até a mesa1, ou mesa2, dos berços de usinagem. Quando o *pallet* chegar à mesa, o robô deve pegar a peça e conduzi-la até o torno, fresadora ou CNC R1, conforme determinado pelo controle computacional. Após carregar o CNC, o robô deve ficar numa posição de repouso pré-determinada ("Phome") aguardando a próxima solicitação de movimentação. Ao término da usinagem, o robô retira a peça pronta do interior do equipamento CNC e a devolve ao *pallet* na mesa1 ou mesa2. Nesse momento o armazém automático encaminha o *pallet* para a armazenagem ou saída, conforme a estratégia de movimentação. Uma análise completa e pormenorizada de todos os eventos da célula é abordada no capítulo 5 (integração da célula).

4.2 INTERFACES

Para realizar a integração da FMC é necessário inicialmente que os diferentes equipamentos troquem dados entre si. Para Vieira (1996), o maior problema na integração de sistemas de manufatura reside na necessidade de interconectar dispositivos heterogêneos, com padrões diversos e produzidos por fabricantes diferentes. Esta afirmação é completamente pertinente para a célula em questão. O torno e o centro de usinagem (fresadora) não possuem nenhum meio de comunicação que permita

alguma interatividade com o meio externo. Ambos CNCs possuem apenas uma porta serial RS232 que é utilizada somente para efetuar o carregamento de programas de usinagem. O controlador do robô possui um cartão Profibus e uma porta serial RS232 para comunicação com dispositivos externos, mas não possui os protocolos necessários para implementação de aplicativos que interajam com o meio externo. Os protocolos são proprietários, gerados somente pelo fabricante do equipamento, e apresentam custos elevados. O armazém automático é o dispositivo melhor preparado para a integração, pois possui uma IHM (Interface Homem-Máquina) desenvolvida a partir de um supervisório SCADA.

A estratégia utilizada para interconectar os diversos equipamentos, ou recursos produtivos, foi a construção de um gerenciador para cada equipamento, utilizando o *software* supervisório Elipse SCADA. Os diferentes gerenciadores trocam mensagens entre si utilizando uma rede de comunicação *Ethernet*. Cada gerenciador troca informações com seu respectivo equipamento através de CLPs. A figura 4.2 apresenta um resumo dessa arquitetura.

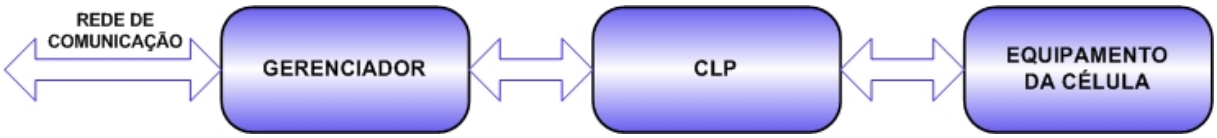


Figura 4.2. Conexão entre gerenciador, CLP e equipamento da FMC.

Para iniciar a modelagem da célula é necessário definir o conjunto de eventos possíveis, a hierarquia do sistema de controle e o número mínimo de mensagens necessárias para estabelecer o sincronismo das ações na FMC.

A definição das mensagens necessárias para a modelagem da célula está alicerçada na fusão de dois tópicos principais. O primeiro deles se refere à escolha da arquitetura de controle heterárquica, que descentraliza o processo de decisão e aumenta a autonomia do sistema. Nesta arquitetura de controle a tomada de decisão é local e autônoma (TEIXEIRA, 2006). Entre os diferentes gerenciadores construídos para a integração da célula nenhum deles possui um nível hierárquico superior ao outro. Todos os elementos de controle (gerenciadores) atuam no mesmo nível e de forma cooperativa. O segundo ponto se refere ao uso do controle modular de Sistemas a Eventos Discretos (SEDs), proposto inicialmente por Wonham e Ramadge, que permite que problemas complexos possam ser decompostos em módulos mais simples, para depois voltar a montar as soluções numa estrutura

modular (WONHAM e RAMADGE, 1989). Torrico e Cury (2004) propõem um aprofundamento deste modelo num controle supervisório hierárquico modular por agregação de estados. Neste controle a hierarquia é dividida em dois níveis, um nível associado ao operador e um outro nível associado ao gerente. A aplicação desta arquitetura permite um comportamento consistente e não bloqueante, entre os níveis de hierarquia e a ação conjunta dos diversos gerenciadores no nível baixo.

Para definir as interfaces mínimas necessárias para a integração de cada equipamento da célula no presente trabalho, utiliza-se a filosofia de controle modular, decompondo horizontalmente as ações possíveis para cada equipamento. Desta forma é possível desmembrar a totalidade de eventos em um conjunto de ações. Internamente cada gerenciador estabelece dois níveis de hierarquia. O primeiro nível, associado ao gerente, está representado pelo gerenciador do equipamento e o conjunto de condições e regras que compõem o processo decisório. O segundo nível, que está associado ao operador, são os elementos de execução, que são responsáveis pela efetiva integração dos gerenciadores com os equipamentos, realizada através de CLPs.

4.2.1 Interfaces dos Equipamentos CNC

Uma análise prévia do *layout* da célula indica que o robô é o responsável pelo carregamento e descarregamento de peças no torno CNC, centro de usinagem CNC e CNC R1. O gerenciador da célula é quem deve informar cada equipamento CNC qual programa de usinagem deve ser carregado. Não há troca de informações entre o armazém automático e os equipamentos CNC.

Após decompor horizontalmente todas as ações possíveis dos equipamentos CNC dentro da célula, e levando em consideração que cada módulo gerenciador deverá ter a maior autonomia possível, determinaram-se as interfaces mostradas na figura 4.3.

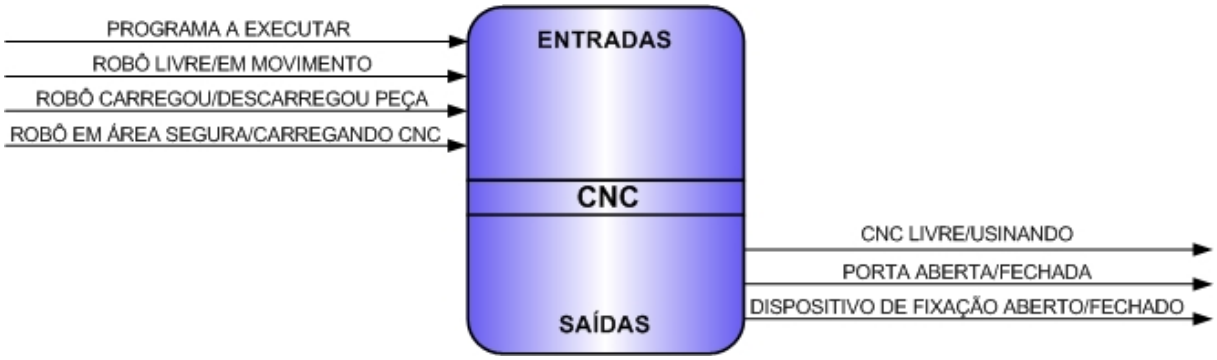


Figura 4.3. Interfaces dos equipamentos CNC

Os CNCs devem receber as informações a respeito de que programa deve ser executado, proveniente do gerenciador da célula. As informações sobre os estados do robô são fornecidas diretamente pelo gerenciador do robô e indicam se o robô está livre ou em algum procedimento de carga ou descarga, se colocou ou retirou alguma peça no dispositivo de fixação das máquinas CNC e, finalmente, se o robô se encontra dentro de uma área considerada segura para operação das máquinas CNC, evitando risco de colisão.

As informações que os CNCs devem fornecer ao sistema, neste caso ao gerenciador da célula e ao gerenciador do robô, se resumem a indicar se a máquina CNC está livre ou executando um processo de usinagem, se a porta está aberta ou fechada e se o dispositivo de fixação automático (castanha, morsa, pinça, etc.) está aberto ou fechado.

Com estas interfaces definidas e padronizadas, é possível incluir um número significativo de diferentes equipamentos CNC na FMC, desde que estas se encontrem fisicamente dentro da área de trabalho do robô. A forma com que os equipamentos CNC fornecem estas informações ao sistema será abordada com detalhes no capítulo 5 desta dissertação.

4.2.2 Interfaces do Robô

Utilizando ainda a filosofia de decompor horizontalmente as ações do robô dentro da FMC, e considerando que o robô movimenta as peças entre o armazém automático e os equipamentos CNC para usinagem, e realiza o caminho inverso ao fim do processamento, pode-se definir o conjunto de interfaces conforme apresentado pela figura 4.4.

Todos os equipamentos CNC (torno, fresadora e CNC R1) devem fornecer ao robô indicações sobre se estão livres ou usinando, a condição de suas portas e seus dispositivos de fixação. O armazém automático informa ao robô sobre a presença de peças a serem usinadas na mesa1 ou na mesa2 dos berços de usinagem. O gerenciador da FMC deve indicar qual programa o robô deve executar.

O robô precisa informar sua condição de livre ou em movimentação de carregamento, a posição para carregamento ou descarregamento nos dispositivos de fixação dos equipamentos CNC, a retirada ou devolução de peças no *pallet* presente na mesa1 ou na mesa2 dos berços de usinagem, a presença dentro dos limites considerados como área segura de operação e a condição de anormalidade operacional ou emergência. Os receptores das respectivas mensagens, assim como uma análise mais

detalhada da troca de informações entre os componentes da FMC serão analisados em detalhes no capítulo 5.

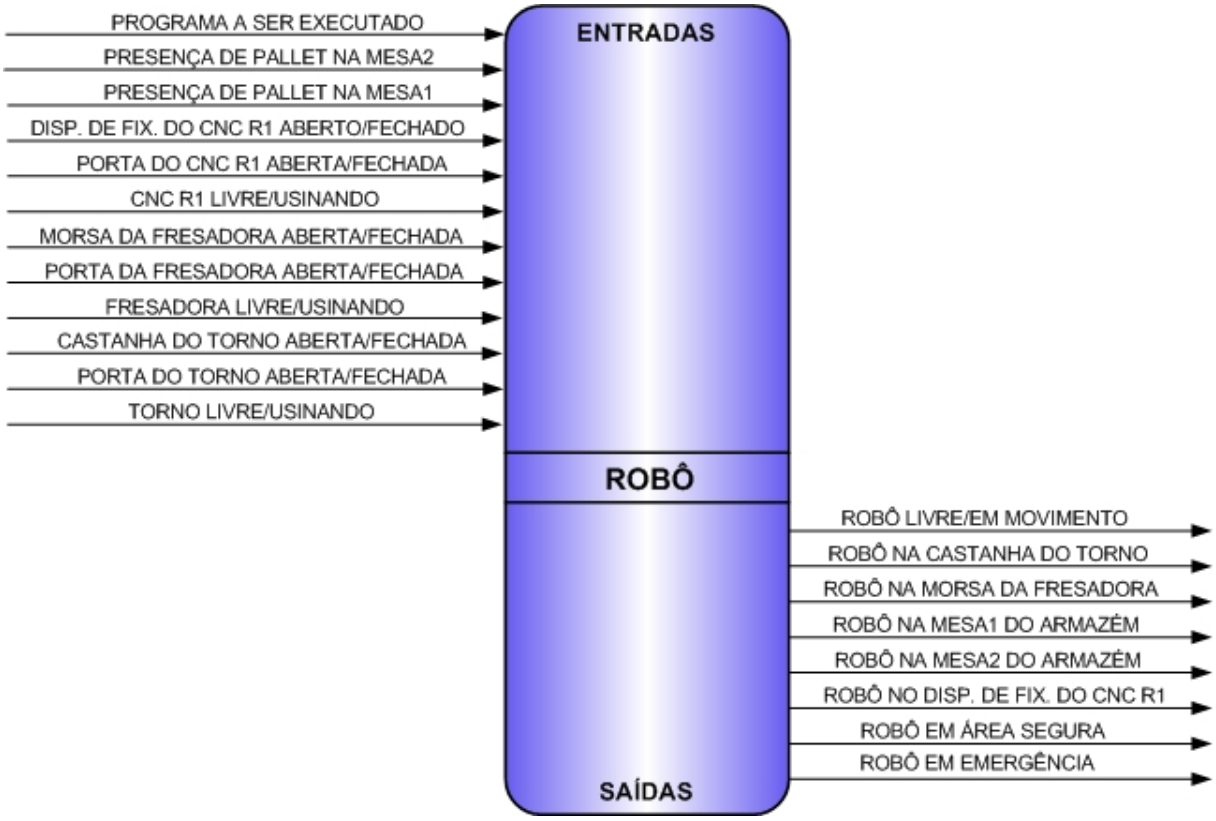


Figura 4.4. Interfaces do robô.

Com este conjunto de interfaces, o robô está apto a fornecer e receber as informações necessárias para carregar os programas apropriados e sincronizar todos os eventos de manipulação de material na FMC.

4.2.3 Interfaces do Armazém Automático

O armazém automático é o responsável inicialmente por fornecer a matéria-prima a ser processada. Após a usinagem, o armazém pode estocar o produto pronto ou encaminha-lo à esteira de saída, para posterior transporte e entrega ao cliente final. A decomposição horizontal das ações possíveis no armazém é mostrada na figura 4.5.

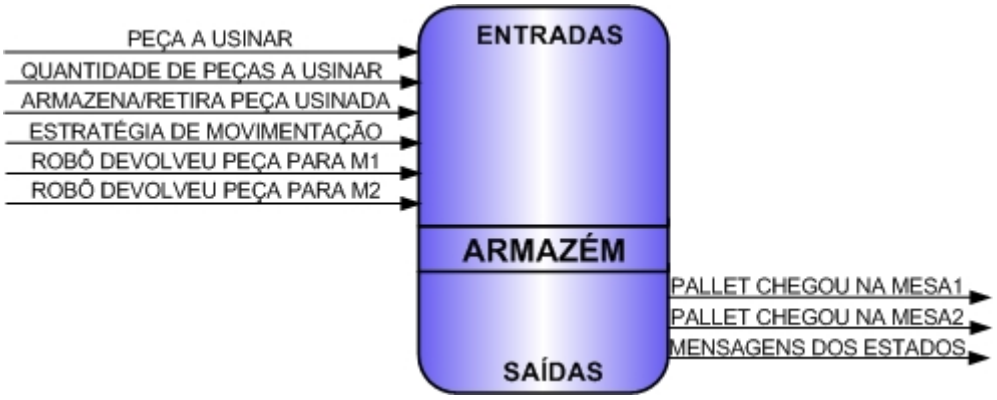


Figura 4.5. Interfaces do armazém automático.

O armazém automático recebe informações do gerenciador FMC e do robô. As informações a respeito de quais peças foram selecionadas para usinagem, a quantidade e o destino do produto já processado, assim como a estratégia de movimentação do transelevador, são recebidos do gerenciador FMC. O retorno das peças, provenientes dos equipamentos CNC, para a mesa1 ou para a mesa2, é sinalizado pelo robô.

As informações que o armazém automático precisa disponibilizar ao sistema resumem-se à chegada do *pallet*, com peças a serem processadas, na mesa1 ou mesa2 dos berços de usinagem, e às mensagens dos estados internos do armazém e do transelevador.

Além das interfaces descritas até o momento, existem outras associadas ao armazém como, por exemplo, a entrada e a estocagem de matéria-prima. Estas interfaces não são analisadas à luz da decomposição das ações possíveis do armazém, para determinar um pacote mínimo de informações, porque elas não são essenciais para o sincronismo e integração da célula. Essas interfaces são responsáveis pela troca de dados entre o gerenciador do armazém e seu banco de dados, ou seja, são informações tratadas de forma autônoma pelo gerenciador do armazém, diminuindo sensivelmente o tráfego de mensagens entre os diferentes gerenciadores da célula.

4.2.4 Interfaces do Gerenciador FMC

O gerenciador FMC tem a função primária de servir de interface com o operador local e remoto, permitindo que os mesmos insiram pedidos de fabricação de peças, indiquem a estratégia de movimentação, prioridade, quantidade, etc. Secundariamente o gerenciador deve monitorar os principais eventos associados aos equipamentos da célula, determinar e comunicar a escolha dos

programas de usinagem e de movimentação do robô. Hierarquicamente o gerenciador FMC se encontra no mesmo nível que os outros gerenciadores da célula. A decomposição horizontal das ações relevantes para o gerenciador FMC é o resumo dos eventos mais importantes dos demais gerenciadores, e pode ser visto na figura 4.6.

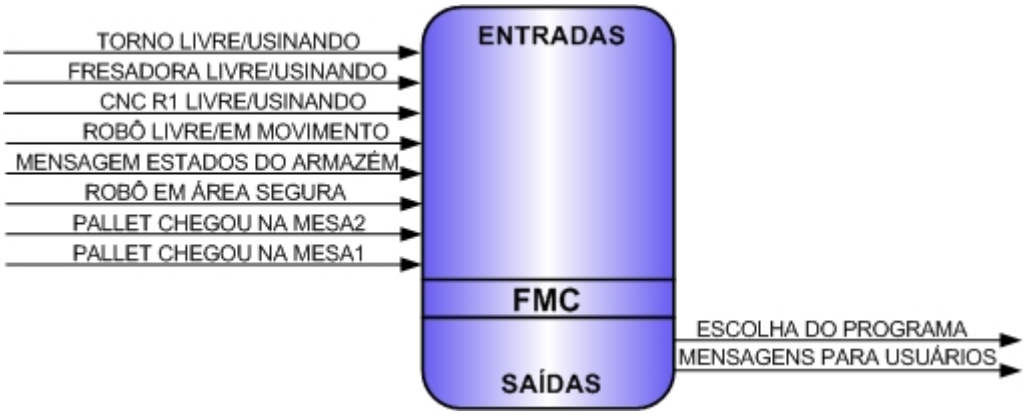


Figura 4.6. Interfaces do gerenciador FMC.

4.3 MODELAGEM A EVENTOS DISCRETOS - REDES DE PETRI

Para efetuar a modelagem de uma Célula Flexível de Manufatura, várias ferramentas podem ser utilizadas, entre elas os Autômatos Finitos, ou Máquinas de Estado; as Redes de Petri; e o SFC (*Sequential Function Chart*), também conhecido como Grafcet (*GRAphe Fonctionnel de Commande Etape/Transition*) (MALER, 1999).

Para Friedrich (1996), o formalismo das Redes de Petri e o formalismo dos Autômatos Finitos são considerados métodos baseados em modelos orientados a transição que seguem uma abordagem operacional. Neste caso, a abordagem operacional descreve o sistema a partir de um modelo executável. O modelo pode ser matematicamente verificado através do uso de análise estática e validado a partir de sua execução. As duas abordagens (RdP e Autômatos Finitos) são aplicáveis à modelagem de sistemas de manufatura; entretanto, as redes de Petri apresentam uma expressividade maior que os Autômatos, ou máquina de Estados. Segundo Mušič e Matko (1998), um dos maiores problemas no uso de Autômatos para o desenvolvimento da estrutura dos sistemas de supervisão e controle em processos industriais reais, é o problema da explosão de estados. Neste sentido as RdP se

destacam, especialmente no que se refere à possibilidade de modelagem de aspectos referentes à sincronização e ao não-determinismo da execução, ou paralelismo, e por apresentar uma estrutura adicional de informações que reduz o problema da explosão de estados.

O SFC, ou Grafcet, é uma linguagem gráfica que proporciona uma representação em forma de diagrama da modelagem de um sistema a eventos discretos. Ela foi definida como padrão para a programação de CLPs pela norma IEC 1131-3 (MALER, 1999). O Grafcet herdou muitas das características da teoria das RdP, e evoluiu como uma simplificação desta na modelagem de processamento paralelo. O comportamento das entradas e saídas no Grafcet é semelhante ao comportamento das entradas e saídas nas RdP, segundo Mušič e Matko (1998). Os autores também apresentam uma metodologia para conversão segura de sistemas modelados em RdP para SFC. Lino e Gomes (2005) apresentam outra metodologia que sugere utilizar inicialmente as potencialidades das RdP para modelagem e verificação formal das propriedades do modelo, e posterior tradução para SFC.

Nesta dissertação, a ferramenta escolhida para efetuar a modelagem da Célula Flexível de Manufatura foi o mecanismo formal de descrição das Redes de Petri. Esta escolha foi feita em razão de existir uma boa base bibliográfica que recomenda seu uso em sistemas a eventos discretos, como é o caso dos Sistemas Flexíveis de Manufatura (TENG e ZHANG, 1993), (VIEIRA, 1996), (MUŠIČ e MATKO, 1998), (GUSTIN, 1999), (ZHOU e VENKATESH, 2000), (RIBEIRO e ELVAS, 2004), (LINO e GOMES, 2005), (MARRANGHELLO, 2005) e (TEIXEIRA, 2006). Além disso, as RdPI descrevem em detalhes as características das condições necessárias para o disparo de cada transição, e a correspondente execução das ações necessárias para o funcionamento do sistema. Esta descrição de condições e ações é transposta de forma quase literal para a geração dos códigos de controle supervisório no *software* Elipse SCADA, facilitando sobremaneira a construção dos gerenciadores.

As Redes de Petri (RdP) foram criadas a partir da apresentação da tese de doutorado de Carl Adam Petri (*Kommunication mit Automaten* – Comunicação com Autômatos) na Universidade de Darmstadt, na Alemanha, em 1962. As RdP originalmente foram utilizadas para representar sistemas a eventos discretos.

Os elementos básicos que permitem a definição de uma RdP são: Lugar, Transição e Ficha (CARDOSO e VALETTE, 1997):

- O lugar é representado por um círculo e pode indicar uma condição, uma espera, um estado parcial ou um procedimento. Normalmente há um predicado associado a cada lugar, por exemplo: “robô em movimento”;

- As transições são representadas por um retângulo e indicam um evento ou ação que ocorre no sistema, por exemplo: “fechar a castanha do torno”;
- As fichas são representadas por um ponto, dentro de um lugar, e significam que a condição associada ao lugar é verificada, ou seja, que o predicado associado ao lugar é verdadeiro. Por exemplo, uma ficha no lugar “robô em movimento” significa que o robô está efetivamente em movimento.

Os estados de uma RdP (lugares) representam a parte estática do sistema modelado, ou seja, todos os estados elementares pelos quais o sistema pode passar em alguma fase do seu funcionamento. As ações (transições) representam a parte dinâmica do sistema modelado, isto é, cada operação que o sistema pode executar durante seu funcionamento. Lugares e transições são vértices do grafo associados às RdP e estão ligados por meio de arcos direcionados. Os arcos que ligam lugares a transições, representam relações entre condições verdadeiras e possibilitam que ações sejam executadas dentro do sistema (MARRANGHELLO, 2005).

Formalmente uma Rede de Petri ordinária é dada por uma quádrupla:

$$N = (P, T, Ppre, Ppost)$$

Onde:

P – {p1, p2,..., pn} é o conjunto finito de lugares da rede N;

T – {t1, t2,..., tn} é o conjunto finito de transições da rede N;

Ppre e *Ppos* $\in IN^{|P| \times |T|}$ são as matrizes de pré e pós-incidência da rede N.

A análise e interpretação das propriedades de uma RdP permitem observar as principais características do sistema que está sendo modelado, e identificar a presença, ou ausência, de propriedades funcionais específicas. Entre as propriedades que dependem do estado inicial, ou da marcação da rede, estão:

- Alcançabilidade – demonstra a capacidade do sistema de atingir um determinado estado, permitindo descrever o comportamento dinâmico do sistema através de um mapa de estados alcançáveis;
- Vivacidade – verifica a potencialidade da rede em disparar em todas as marcas alcançáveis. Uma rede é dita viva quando não apresenta *deadlocks* (bloqueios mortais);

- Reversibilidade – é a capacidade da rede em retornar à marcação inicial a partir de qualquer marcação;
- Conflito – é a existência, num determinado estado, de duas possibilidades excludentes de evolução, ocasionando indeterminações no sistema modelado.

Segundo Zhou e Venkatesh (2000), a RdP é uma ferramenta matemática e gráfica que oferece um ambiente uniforme para modelagem, análise e projeto de sistemas a eventos discretos. Ainda segundo os autores, algumas das características que fazem das RdP uma poderosa ferramenta para implementação de sistemas automatizados de manufatura são: as facilidades para modelar as características de complexos sistemas industriais; a modelagem em RdP oferece excelente visualização das relações de dependência entre recursos; facilidade para gerar os códigos de controle supervisório diretamente da representação gráfica oferecida pelas RdP; capacidade de prever os indesejados *deadlock* no sistema; a capacidade de modelar e simular Sistemas a Eventos Discretos (SEDs). Cury (2001) define SEDs como "... um sistema dinâmico que evolui de acordo com a ocorrência abrupta de eventos físicos, em intervalos de tempo em geral irregulares e desconhecidos".

De acordo com Teng e Zhang (1993), o uso de RdPs se adapta muito bem ao controle de complexos sistemas de manufatura, ou sistemas dinâmicos a eventos discretos, como é o caso dos Sistemas Flexíveis de Manufatura - FMS. As RdPs fornecem modelos fiéis e métodos eficientes de análise, identificam interações de eventos paralelos e seqüenciais e permitem a implementação de análise em tempo real.

Na modelagem da célula foi utilizada a Rede de Petri Interpretada (RdPI). A característica das RdPI é a de apresentar dois grupos de parênteses associados a cada transição. O primeiro grupo indica uma condição a ser satisfeita antes do disparo da transição. Se a transição estiver sensibilizada, porém sem atender a condição imposta, a transição não ocorre. Neste caso, diz-se que a transição está sensibilizada, porém não está habilitada. O segundo grupo de parênteses representa as ações decorrentes do disparo da transição. A figura 4.7 apresenta a estrutura da RdPI que modela e interpreta a movimentação da matéria-prima da mesa1 do berço de usinagem até o torno. Os lugares representam os estados dos recursos; as transições modelam a seqüência de ações (eventos) que modificam os estados dos recursos; os arcos direcionados representam o fluxo do processo; as fichas indicam que o predicado associado ao lugar é verdadeiro. A volta da peça já processada pelo torno para o *pallet* presente na mesa1 do berço de usinagem é descrita por outra RdPI. Todas as RdPI desenvolvidas na modelagem da célula são apresentadas no Apêndice A desta dissertação.

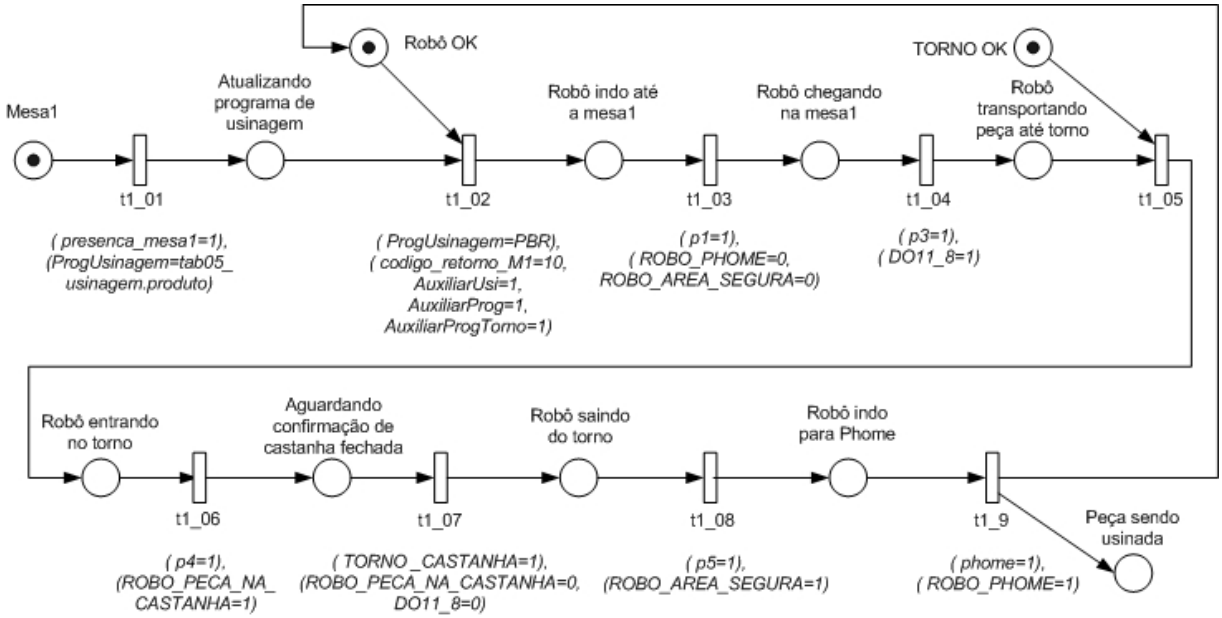


Figura 4.7. RdPI simplificada para movimentação de peças da mesa1 até o torno.

Uma análise preliminar dessa rede de Petri indica que ela é composta por:

- 12 lugares: Mesa1
  Atualizando programa de usinagem
  Robô OK
  Robô indo até a mesa1
  Robô chegando na mesa1
  Robô transportando peça até torno
  TORNO OK
  Robô entrando no torno
  Aguardando confirmação de castanha fechada
  Robô saindo do torno
  Robô indo para Phome
  Peça sendo usinada
- 9 transições: t1_01; t1_02; t1_03; t1_04; t1_05; t1_06; t1_07; t1_08; t1_09.
- 16 variáveis: presença_mesa1 (presença de *pallet* na mesa1)
  ProgUsinagem (programa de usinagem)
  codigo_retorno_M1 (código de retorno da peça para a mesa1)
  AuxiliarUsi (código do programa de usinagem)

AuxiliarProg (código do programa do robô)

AuxiliarProgTorno (código de programa para torno)

p1 (ponto 1 na trajetória do robô)

ROBO_PHOME (robô livre)

ROBO_AREA_SEGURA (robô dentro da área de segurança)

p3 (ponto 3 na trajetória do robô)

DO11_8 (acionamento da garra do robô)

p4 (ponto 4 na trajetória do robô)

ROBO_PECA_NA_CASTANHA (robô coloca ou retira peça na castanha do torno)

TORNO_CASTANHA (castanha do torno aberta ou fechada)

p5 (ponto 5 na trajetória do robô)

phome (ponto phome na trajetória do robô)

Esta rede apresenta a movimentação de uma peça a ser usinada e que chegou à mesa1 do berço de usinagem. Se a condição *presença_mesa1=1* for atendida, ou seja, se houver um *pallet* contendo matéria-prima na mesa1, então a transição t1_01 dispara executando a ação *ProgUsinagem=tab05_usinagem.produto*, que atualiza a variável *ProgUsinagem* com o conteúdo da coluna produto, da tabela de usinagem (número 05) do banco de dados. Esta ação ocasionará a atualização do programa de usinagem, representado pelo lugar *Atualizando programa de usinagem*. A transição t1_02 apresenta a condição de disparo *ProgUsinagem=PBR*. Se o programa de usinagem carregado for igual ao código PBR (Peão Branco, como ilustrado na tabela 3.1), a transição dispara executando *código_retorno_M1=10* (significa que a peça será encaminhada ao torno), *AuxiliarUsi=1* (carrega o programa número 1 no gerenciador do torno), *AuxiliarProg=1* (carrega o programa número 1 no robô) e *AuxiliarProgTorno=1* (carrega a combinação BCD igual a 1 no CLP do torno). O disparo da transição 2 inicia o movimento do robô para buscar a matéria-prima na mesa1. A transição t1_03 somente dispara quando o robô chega ao ponto p1 (*p1=1*). Nesse momento o gerenciador do robô sinaliza que o robô está em movimento (*ROBO_PHOME=0*) e que o robô já não se encontra no interior da área segura (*ROBO_AREA_SEGURA=0*). Quando o robô chega ao ponto p3 (*p3=1*), habilita-se o disparo da transição t1_04, que executa a ação *DO11_8=1* (fecha a garra do robô sobre a peça do *pallet*). Após pegar a peça, o robô leva a mesma até o torno, que está livre e aguardando. O ponto *p4=1*, que habilita o disparo da transição t1_06, significa que o robô colocou a peça na castanha

do torno. Com o disparo da transição também é executada a ação representada por *ROBO_PECA_NA_CASTANHA=1*, que indica a presença de peça na castanha e a conseqüente ordem para fechá-la. O disparo da transição t1_07 somente ocorre quando o torno indica que fechou a castanha, prendendo a peça (*TORNO_CASTANHA=1*). Após a confirmação do fechamento da castanha e o conseqüente disparo da transição, são executados *TORNO_CASTANHA=0* (encerra a ordem para fechamento da castanha) e *DO11_8=0*, que abre a garra do robô. Cumprida a etapa de carregamento do torno, o robô sai do interior do CNC e quando alcança o ponto p5 (*p5=1*), dispara a transição t1_08. Como resultado do disparo da transição é executada a ação *ROBO_AREA_SEGURA=1*, que informa que o robô se encontra novamente em área segura, permitindo o fechamento da porta do torno e o início da usinagem. A transição t1_09 é disparada quando o robô alcança sua posição de repouso (*phome=1*). Nesse momento o gerenciador do robô indica ao sistema que o robô está livre (*ROBO_PHOME=1*). Dois lugares derivam da última transição: o primeiro indica que a peça está sendo usinada no torno, e o outro disponibiliza o robô para executar outra operação, se houver uma requisição pendente.

4.4 CONVERSÃO DAS RdPI PARA CÓDIGOS DE CONTROLE SUPERVISÓRIO

A partir da modelagem da célula utilizando as RdPI, é possível traduzir as condições de disparo de cada transição, e as ações decorrentes do disparo, diretamente para os *scripts* de programação do *software* Elipse SCADA. As variáveis utilizadas na modelagem para indicar as condições e as ações associadas a cada transição já estão no formato de *tags*, que serão detalhadas em profundidade no capítulo 5 desta dissertação.

O supervisório Elipse SCADA é um *software* orientado a eventos, o que significa que a execução de qualquer *script* de programação depende diretamente da ocorrência de um fenômeno discreto. Esta característica é utilizada como condição de disparo de cada transição definida na modelagem da célula. O *script* de programação, sempre associado a um evento, contém os códigos que representam as ações a serem executadas pelo sistema. O controle da evolução dos estados no supervisório está centralizado nos eventos que acionam os disparos das transições. A figura 4.8 apresenta um exemplo da conversão da modelagem em RdPI para geração dos controles supervisórios, e representam a atualização dos programas de usinagem da célula, disparada pela chegada de um

pallet na mesa1 dos berços de usinagem do armazém (evento OnValueChanged – dispara sempre que houver mudança de valor da variável).

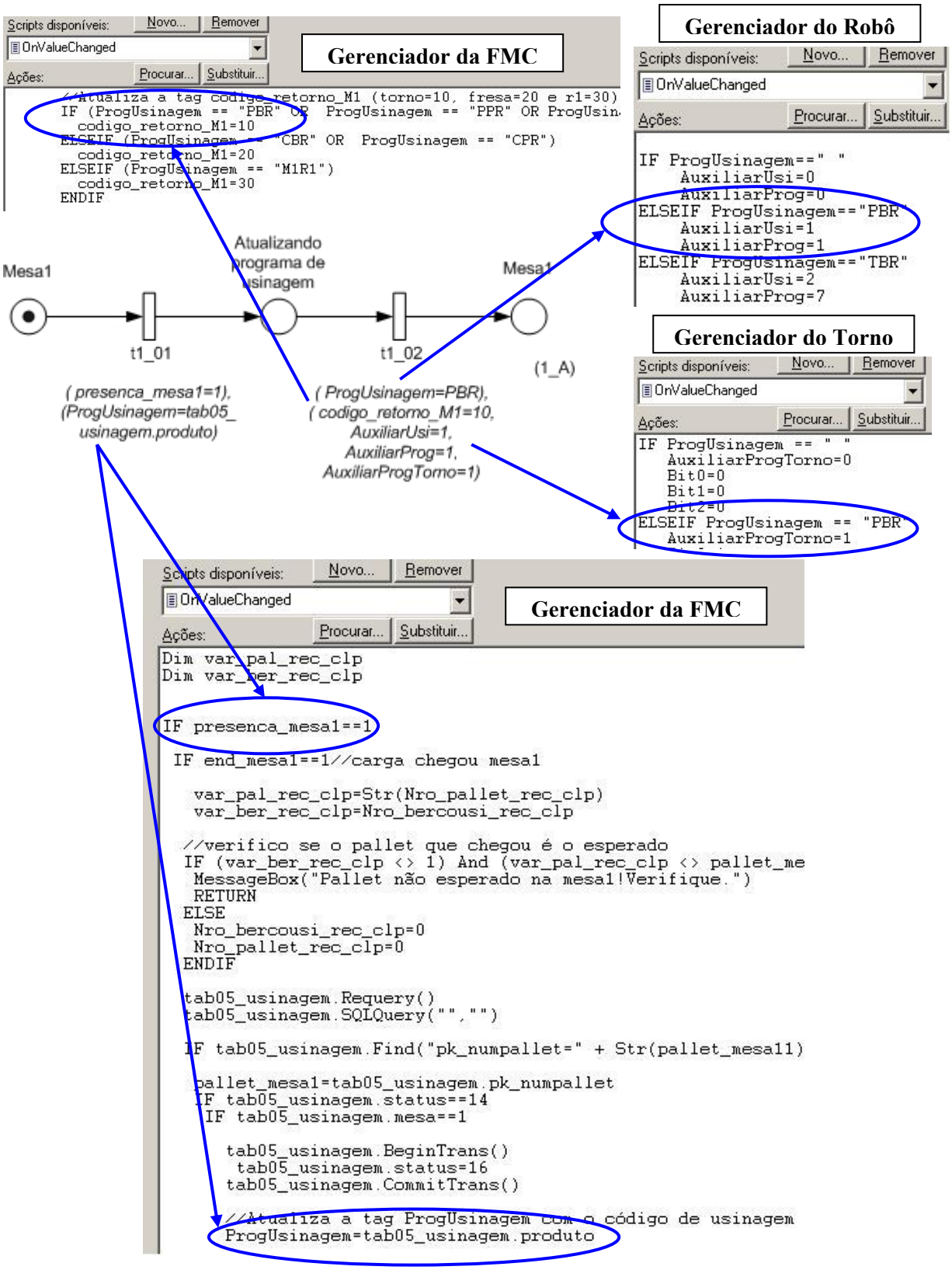


Figura 4.8. Exemplo de conversão da RdPI para o código de controle

5. INTEGRAÇÃO DA CÉLULA

Este capítulo trata com detalhes a integração da FMC, a partir da modelagem da célula por RdPI, a definição de um conjunto de interfaces entre os diversos recursos produtivos e a definição de uma arquitetura, que utiliza gerenciadores para cada equipamento existente na célula. A construção dos gerenciadores e a integração destes com CLPs e respectivos equipamentos (robô, armazém automático e equipamentos CNC) é abordada em detalhes, apresentando as configurações de *tags*, *drivers*, redes e os *softwares* desenvolvidos para CLPs, CNCs e o robô. O *software* utilizado para a construção dos gerenciadores da célula é o supervisório Elipse SCADA, versão v2.28 build 024.

5.1 ARQUITETURA DA CÉLULA

A arquitetura proposta para a integração da célula está baseada na construção de gerenciadores para cada recurso produtivo da célula, mais um gerenciador da FMC e um gerenciador remoto. Os gerenciadores trocam informações por meio de uma rede local *Ethernet*, utilizando o padrão de comunicação OPC. Todos os gerenciadores são servidores e clientes OPC. Os gerenciadores dos equipamentos CNC (torno, fresadora e CNC R1) estão conectados a seus respectivos equipamentos por meio de CLPs, assim como o gerenciador do robô. Já o gerenciador do armazém automático está conectado diretamente a um CLP *master*, que por sua vez utiliza um CLP secundário, denominado *slave*.

O conjunto de mensagens trocadas entre os gerenciadores, assim como entre os gerenciadores e os respectivos CLPs, foram previamente definidas na modelagem da célula. A figura 5.1 exemplifica a arquitetura proposta para a integração da célula.

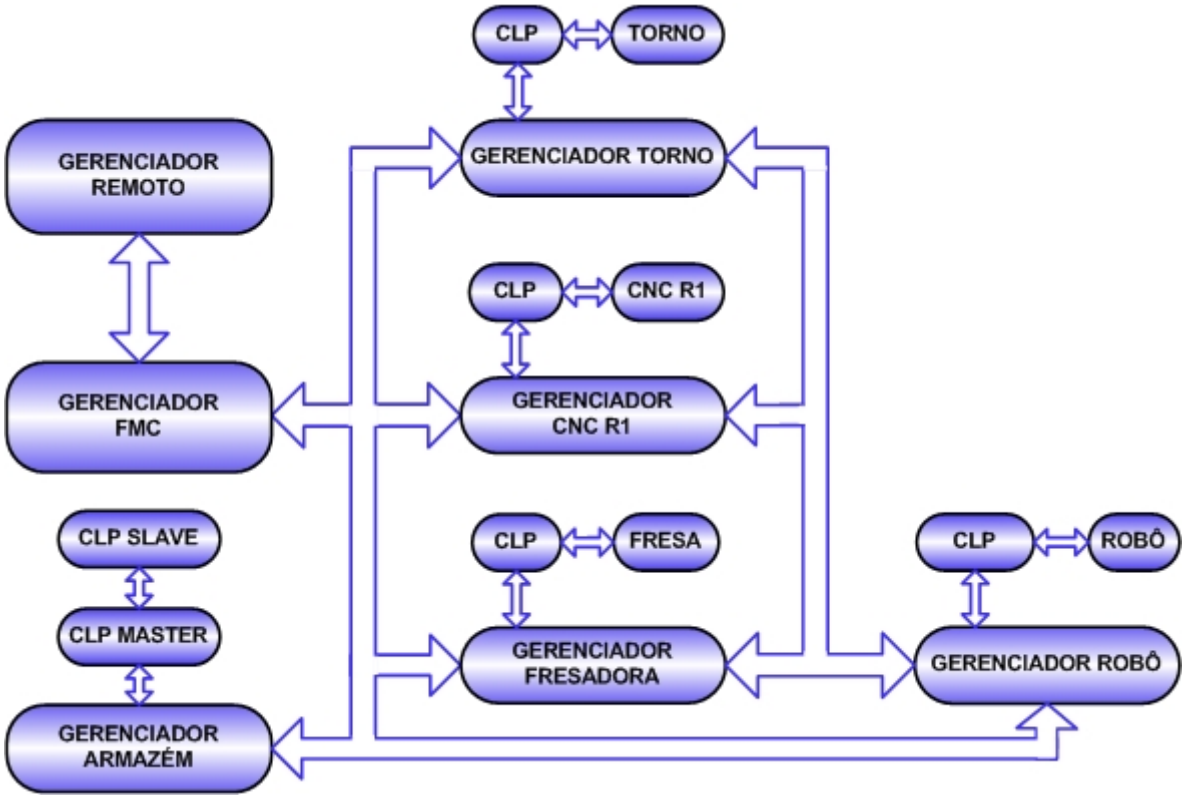


Figura 5.1. Arquitetura da FMC.

5.2. GERENCIADOR DO TORNO

A construção do gerenciador dos equipamentos CNC (torno, centro de usinagem e CNC R1) está baseada integralmente na função que os mesmos desempenharão na célula de manufatura e na determinação das interfaces de cada equipamento. A modelagem utilizando RdPI permite visualizar o comportamento dinâmico de cada CNC na célula, os eventos e os sincronismos necessários. Parte-se do princípio de que os equipamentos CNC devem ter a maior autonomia possível. Desta forma os programas de usinagem deverão ser previamente carregados e identificados. O gerenciador indicará qual é o número do programa que o CNC deve carregar.

O supervisório Elipse SCADA trata as variáveis do processo como *tags*. Recuperando as interfaces definidas no capítulo 4.2.1 para os CNCs, e realizando a conversão para *tags* que serão utilizadas pelo torno, temos:

Tabela 5.1. Conversão de interfaces do torno para *tags*

INTERFACES	*TAG*
PROGRAMA A EXECUTAR	ProgUsinagem
ROBÔ LIVRE/EM MOVIMENTO	ROBO_PHOME
ROBÔ CARREGOU/DESCARREGOU PEÇA	ROBO_PECA_NA_CASTANHA
ROBÔ EM ÁREA SEGURA/CARREGANDO CNC	ROBO_AREA_SEGURA
CNC LIVRE/USINANDO	TORNO_OK
PORTA ABERTA/FECHADA	TORNO_PORTA
DISPOSITIVO DE FIXAÇÃO ABERTO/FECHADO	TORNO_CASTANHA

Com exceção da *tag* ProgUsinagem, que assume valores alfanuméricos, as outras *tags* somente podem assumir valores binários. A tabela 5.2 apresenta o significado de cada *tag* conforme o valor assumido.

Tabela 5.2. Significado das *tags* para o gerenciador do torno

TAG	VALOR=0	VALOR=1
ProgUsinagem	Nenhum	Nenhum
ROBO_PHOME	Em movimento	Em Phome (Robô está livre)
ROBO_PECA_NA_CASTANHA	Peça fora da castanha	Peça está na castanha
ROBO_EM_AREA_SEGURA	Robô em área de risco	Robô está em área segura
TORNO_OK	Torno está usinando (OUT)	Torno está livre (OK)
TORNO_PORTA	Porta do torno está fechada	Porta do torno está aberta
TORNO_CASTANHA	Castanha do torno está aberta	Castanha do torno está fechada

A *tag* ProgUsinagem é recebida diretamente do gerenciador FMC e indica através de um valor alfanumérico, qual programa de usinagem previamente cadastrado está sendo solicitado pelo usuário local ou usuário remoto. O gerenciador do robô envia ao gerenciador do torno um conjunto de três mensagens: ROBO_PHOME, ROBO_PECA_NA_CASTANHA e ROBO_AREA_SEGURA. O gerenciador do torno disponibiliza ao sistema também um conjunto de três *tags*: TORNO_OK, TORNO_PORTA e TORNO_CASTANHA.

As informações do gerenciador FMC e do gerenciador do robô são recebidas pelo gerenciador do torno via padrão de comunicação OPC. Neste caso os gerenciadores que fornecem informações se comportam como servidores OPC; já o gerenciador do torno assume o papel de cliente OPC. A figura 5.2 apresenta todas as *tags* utilizadas pelo gerenciador do torno.

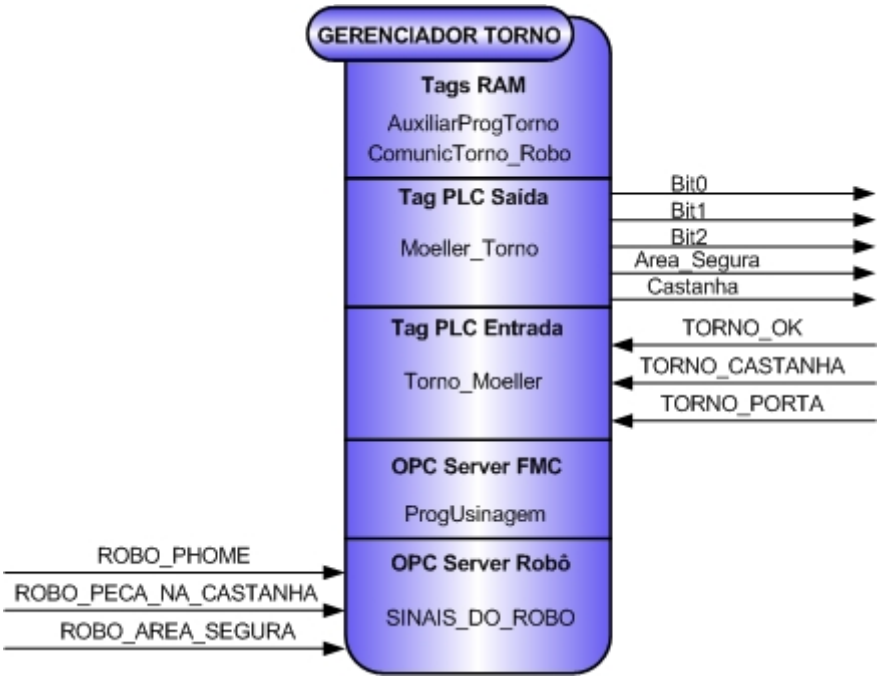


Figura 5.2. *Tags* utilizadas pelo gerenciador do torno.

O conjunto de três *tags* recebidas do servidor OPC do robô é um desmembramento de uma única *tag* OPC denominada SINAIS_DO_ROBÔ. Cada uma das três *tags* desmembradas equivale a um *bit* da *tag* OPC. Este recurso reduz o número de *tags* e, por conseguinte de mensagens, que circulam pela rede de comunicação. A *tag* ProgUsinagem, recebida do servidor OPC da FMC, é decodificado através de um *script* de programação associado à *tag* RAM AuxiliarProgTorno. Para comunicação com o torno, via CLP, são utilizadas duas *tags* PLC, uma para enviar dados, e outra para receber dados do CLP (Torno_Moeller e Moeller_Torno, respectivamente). Para completar o conjunto de *tags* do gerenciador, utiliza-se a *tag* RAM ComunicTorno_robô, para enviar informações dos estados do torno ao robô, via OPC.

5.2.1 Aplicativo gerenciador do torno no Elipse SCADA

A implementação do aplicativo gerenciador do torno, no Elipse SCADA, está baseada na definição das *tags* apresentada pela figura 5.2. Inicialmente são configuradas as *tags* OPC. A figura 5.3 apresenta a configuração do *organizer* do *software* supervisório:

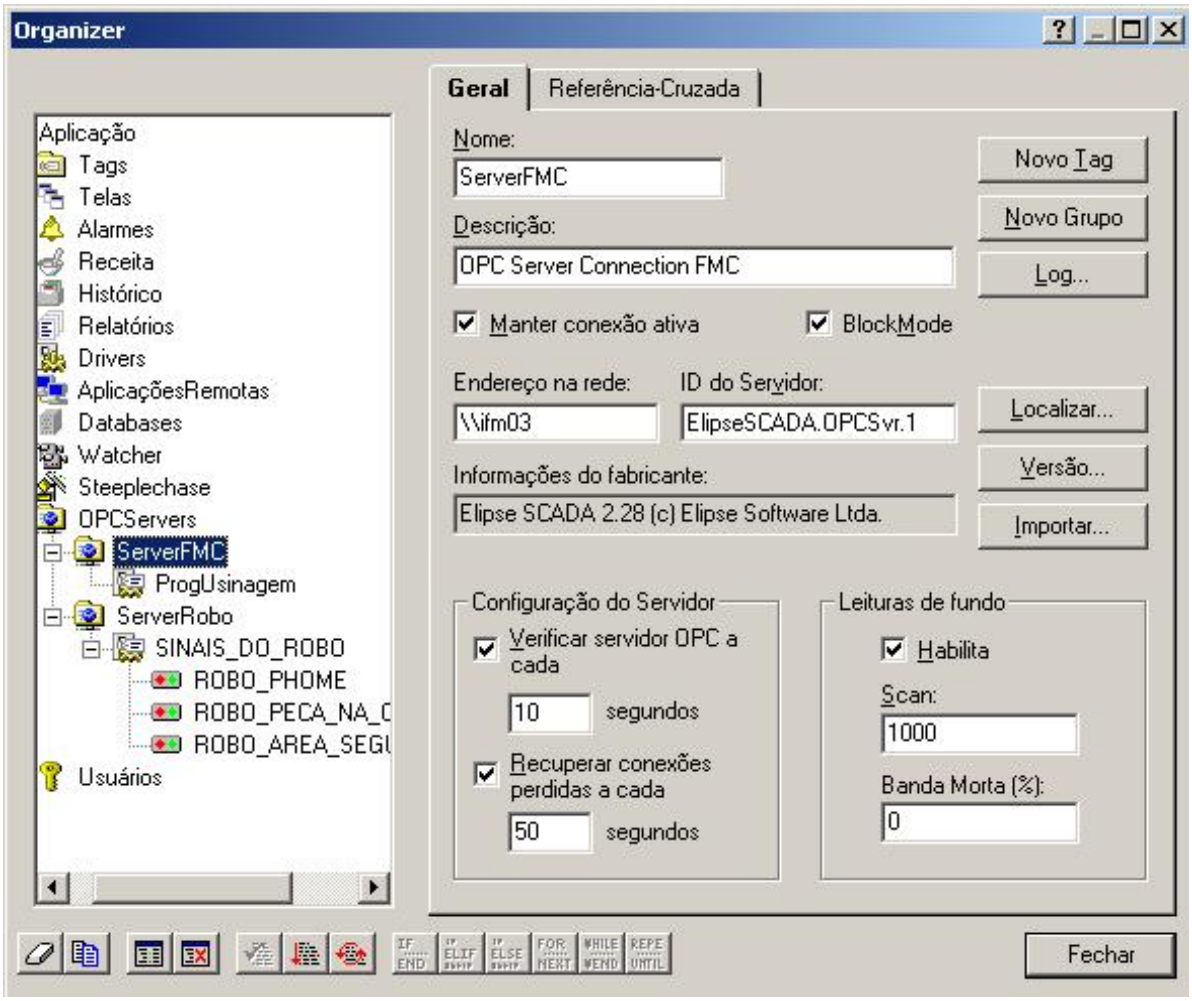


Figura 5.3. Configuração das *tags* OPC no *organizer* do Elipse SCADA.

O gerenciador da FMC, configurado como OPC ServerFMC, teve sua *tag* ProgUsinagem importada do computador "ifm03" da rede. O quadro "ID do Servidor" mostra o servidor OPC selecionado no endereço da rede. Neste caso o gerenciador do torno assume o papel de cliente OPC e o gerenciador FMC, o de servidor OPC. A mesma análise é válida em relação ao gerenciador do robô, configurado como OPC ServerRobo. A diferença está no fato de que a *tag* importada (SINAIS_DO_ROBO) é desmembrada em três outras *tags*: ROBO_PHOME, *bit* 0; ROBO_PECA_NA_CASTANHA, *bit* 1; ROBO_AREA_SEGURA, *bit* 5.

Associado à *tag* ProgUsinagem um *script* decodifica as informações dos códigos recebidos e atualiza as *tags* internas AuxiliarProgTorno, *Bit*0, *Bit*1 e *Bit*2. A execução do *script* está condicionada à ocorrência do evento OnValueChanged, ou seja, cada vez que o conteúdo da *tag* ProgUsinagem sofrer uma atualização, o *script* será executado. Na modelagem da célula por RdPI, a mudança do

valor da *tag* é representativa da condição de disparo da transição, enquanto o *script* é representado pelas ações a serem executadas, em conjunto com o disparo da transição.

Tabela 5.3. *Script* da *tag* ProgUsinagem

Tag ProgUsinagem - *Script* OnValueChanged		
IF ProgUsinagem == " "	ELSEIF ProgUsinagem == "BBR"	ELSEIF ProgUsinagem == "DPR"
AuxiliarProgTorno=0	AuxiliarProgTorno=3	AuxiliarProgTorno=6
Bit0=0	Bit0=1	Bit0=1
Bit1=0	Bit1=1	Bit1=0
Bit2=0	Bit2=0	Bit2=0
ELSEIF ProgUsinagem == "PBR"	ELSEIF ProgUsinagem == "BPR"	ELSEIF
AuxiliarProgTorno=1	AuxiliarProgTorno=4	AuxiliarProgTorno=15
Bit0=1	Bit0=1	Bit0=0
Bit1=0	Bit1=1	Bit1=0
Bit2=0	Bit2=0	Bit2=0
ELSEIF ProgUsinagem == "PPR"	ELSEIF ProgUsinagem == "DBR"	ELSEIF AuxiliarProgTorno=16
AuxiliarProgTorno=2	AuxiliarProgTorno=5	ENDIF
Bit0=1	Bit0=1	
Bit1=0	Bit1=0	
Bit2=0	Bit2=0	

As *tags* OPC ROBO_PECA_NA_CASTANHA e ROBO_AREA_SEGURA também possuem *scripts* associados ao evento OnValueChanged e são mostradas pela tabela 5.4.

Tabela 5.4. *Script* da *tag* SINAIS_DO_ROBO - OnValueChanged

Tag ROBO_PECA_NA_CASTANHA	*Tag* ROBO_AREA_SEGURA
IF ROBO_PECA_NA_CASTANHA==0	IF ROBO_AREA_SEGURA==0
Castanha=0	Area_Segura=0
ELSEIF ROBO_PECA_NA_CASTANHA==1	ELSEIF ROBO_AREA_SEGURA==1
Castanha=1	Area_Segura=1
ENDIF	ENDIF

A configuração das demais *tags* pertencentes ao gerenciador do torno é apresentada pela figura 5.4.

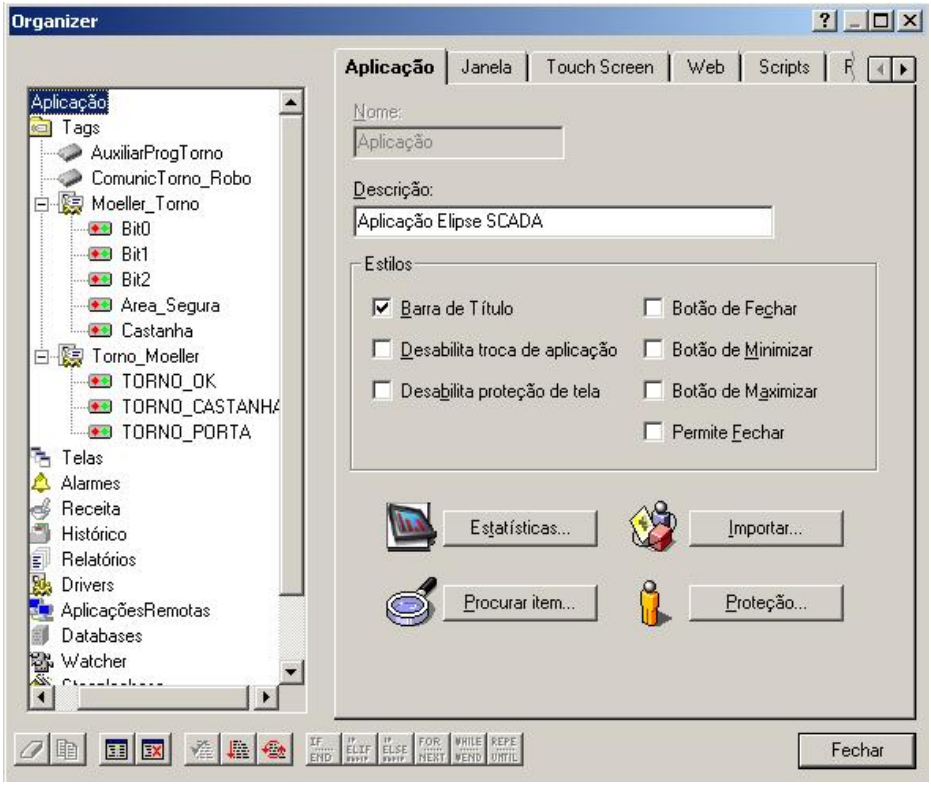


Figura 5.4. *Tags* do gerenciador do torno.

A *tag* RAM AuxiliarProgTorno é a responsável por apresentar ao usuário as mensagens referentes ao processo de usinagem do torno. A *tag* é atualizada pelo *script* associado à *tag* OPC ProgUsinagem, conforme mostrado pela tabela 5.3. A relação entre o código recebido através da *tag* ProgUsinagem, a decodificação efetuada pela *tag* AuxiliarProgTorno e as mensagens apresentadas ao usuário por meio de um objeto tipo texto é a seguinte:

Tabela 5.5. Relação entre *tags* e mensagens ao usuário

ProgUsinagem	AuxiliarProgTorno	Mensagens ao usuário
“ “	0	NENHUM PROGRAMA SELECIONADO PARA
PBR	1	USINAR PEÃO BRANCO
PPR	2	USINAR PEÃO PRETO
BBR	3	USINAR BISPO BRANCO
BPR	4	USINAR BISPO PRETO
DBR	5	USINAR DAMA BRANCA
DPR	6	USINAR DAMA PRETA
EMERG	15	EMERGÊNCIA ACIONADA!
	16	PROGRAMA NÃO É VÁLIDO PARA TORNO!

Para comunicação do gerenciador com o CLP, neste caso um Moeller modelo PS4 201 - módulo MM1, foram configurados as *tags* CLP Moeller_Torno e Torno_Moeller. A primeira tem a

função de enviar ao torno as informações necessárias para a integração com o gerenciador e é desmembrada em: Bit0 (*bit* 0), Bit1 (*bit* 1), Bit2 (*bit* 2), Area_Segura (*bit* 4) e Castanha (*bit* 5). Já a *tag* Torno_Moeller envia as informações do torno para o gerenciador e também é desmembrada em: TORNO_OK (*bit* 8), TORNO_CASTANHA (*bit*9) e TORNO_PORTA (*bit* 10). Um *script* associado ao evento OnValueChanged da *tag* Torno_Moeller atualiza as variáveis dos estados do torno (livre/usinando, porta aberta/fechada e castanha aberta/fechada) assim como da *tag* RAM ComunicTorno_Robo, que será responsável de comunicar dados do torno ao robô, via servidor OPC. O *script* é apresentado pela tabela 5.6.

Tabela 5.6. *Script* da *tag* Torno_Moeller

Tag Torno_Moeller – *Script* OnValueChanged		
IF Torno_Moeller<256	IF Torno_Moeller>=768	IF Torno_Moeller>=1536
TORNO_OK=0	IF Torno_Moeller<1024	IF Torno_Moeller<1792
TORNO_CASTANHA=0	TORNO_OK=1	TORNO_OK=0
TORNO_PORTA=0	TORNO_CASTANHA=1	
ComunicTorno_Robo=0	TORNO_PORTA=0	TORNO_PORTA=1
ENDIF	ComunicTorno_Robo=3	
IF Torno_Moeller>=256	ENDIF	ENDIF
IF Torno_Moeller<512	ENDIF	ENDIF
TORNO_OK=1	IF Torno_Moeller>=1024	IF Torno_Moeller>=1792
	IF Torno_Moeller<1280	IF Torno_Moeller<2048
TORNO_PORTA=0	TORNO_OK=0	TORNO_OK=1
ComunicTorno_Robo=1	TORNO_CASTANHA=0	
ENDIF	TORNO_PORTA=1	TORNO_PORTA=1
ENDIF	ComunicTorno_Robo=4	
IF Torno_Moeller>=512	ENDIF	ENDIF
IF Torno_Moeller<768	ENDIF	ENDIF
TORNO_OK=0	IF Torno_Moeller>=1280	
	IF Torno_Moeller<1536	
TORNO_PORTA=0	TORNO_OK=1	
ComunicTorno_Robo=2	TORNO_CASTANHA=0	
ENDIF	TORNO_PORTA=1	
ENDIF	ComunicTorno_Robo=5	
	ENDIF	
	ENDIF	

A atualização da *tag* CLP Torno_Moeller, e por conseqüência das três *tags* desmembradas, ocorre da leitura de entradas digitais do CLP Moeller. A *tag* ComunicTorno_Robo possui a informação dos estados do torno, transformando a combinação binária das *tags* TORNO_OK (*bit* menos significativo), TORNO_CASTANHA (*bit* intermediário) e TORNO_PORTA (*bit* mais significativo).

A interface gráfica do gerenciador do torno, mostrada na figura 5.5, contém as principais informações que o operador do sistema necessita para identificar de forma fácil e rápida o estado dos equipamentos e sinais que são tratados pelo gerenciador.

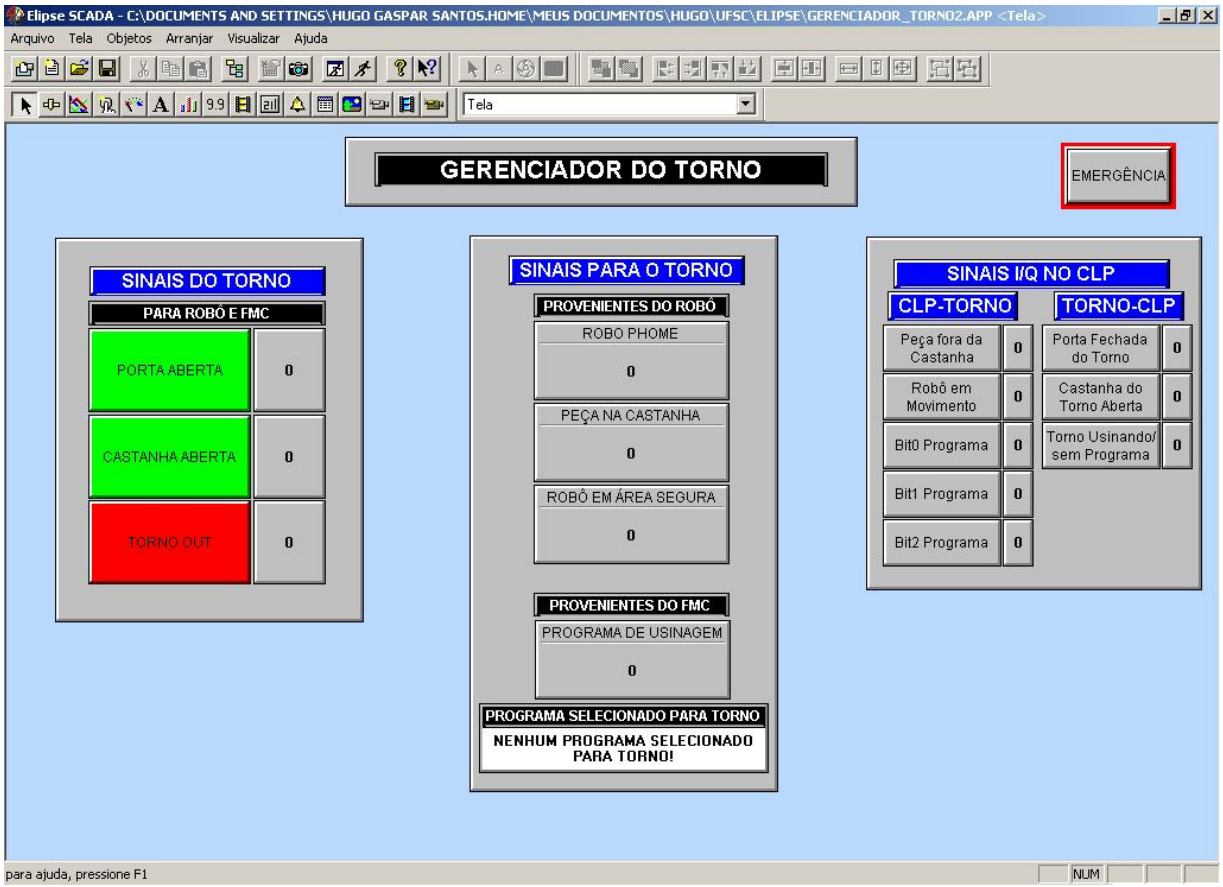


Figura 5.5. Interface com operador do gerenciador do torno

5.2.2 Integração do gerenciador do torno com CLP Moeller

A integração do gerenciador do torno com o CLP Moeller ocorre através das *tags* CLP Moeller_Torno e Torno_Moeller. A comunicação entre o gerenciador e o dispositivo exige que seja configurado um *driver* específico, fornecido pela Elipse, como mostrado na tela do *organizer*, na figura 5.6.

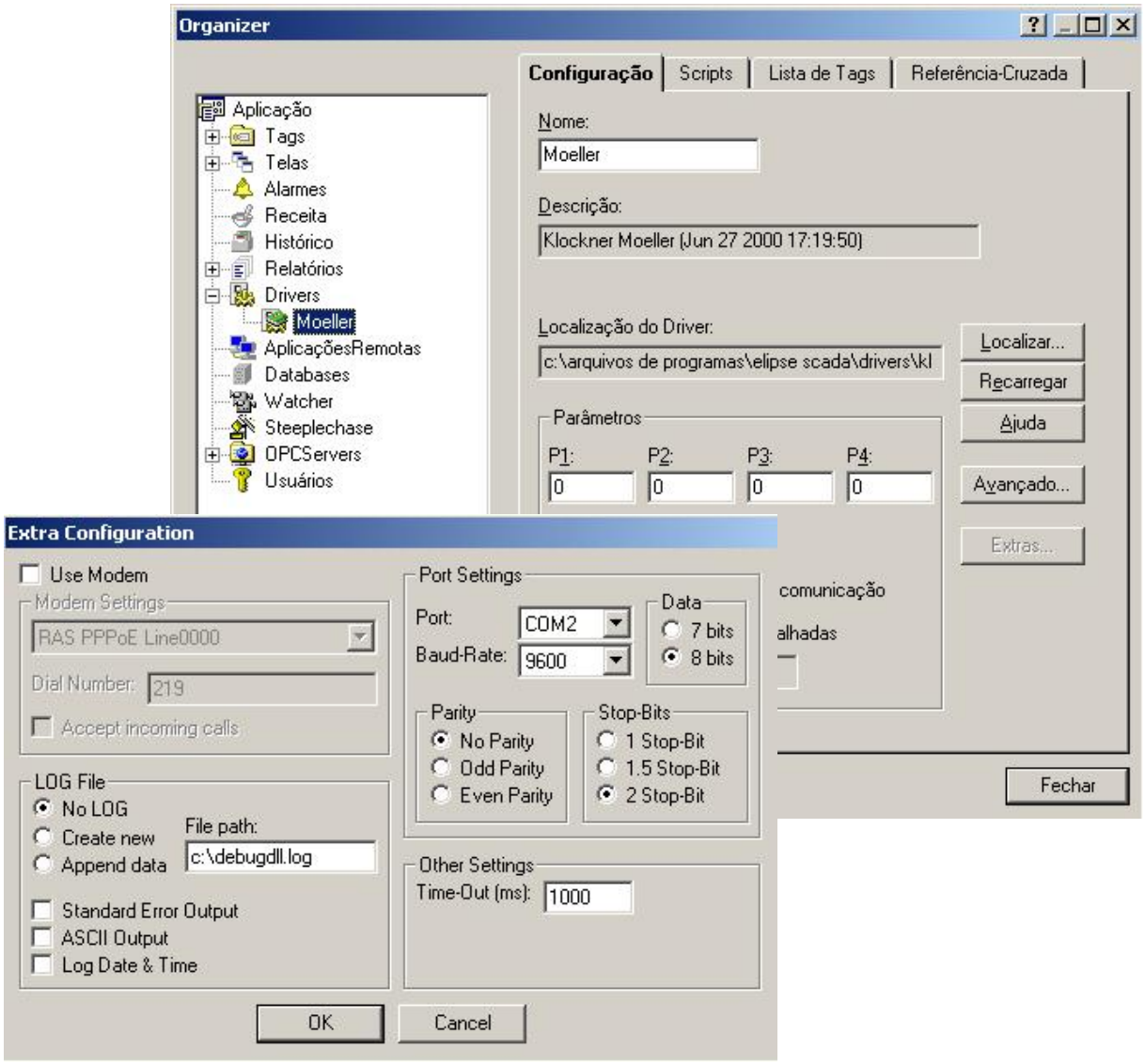


Figura 5.6. *Driver* e configuração do CLP Klockner Moeller.

A comunicação entre gerenciador e o CLP foi configurada para utilizar a porta serial COM2, com velocidade de 9600 bps, 8 *bits* de dados, sem paridade e 2 *bits* de parada. Todos os parâmetros "P" foram deixados com valores nulos (igual a zero), dessa forma a parametrização do *driver* fica por conta dos parâmetros "N", diretamente na *tag* CLP, como mostra a figura 5.7.

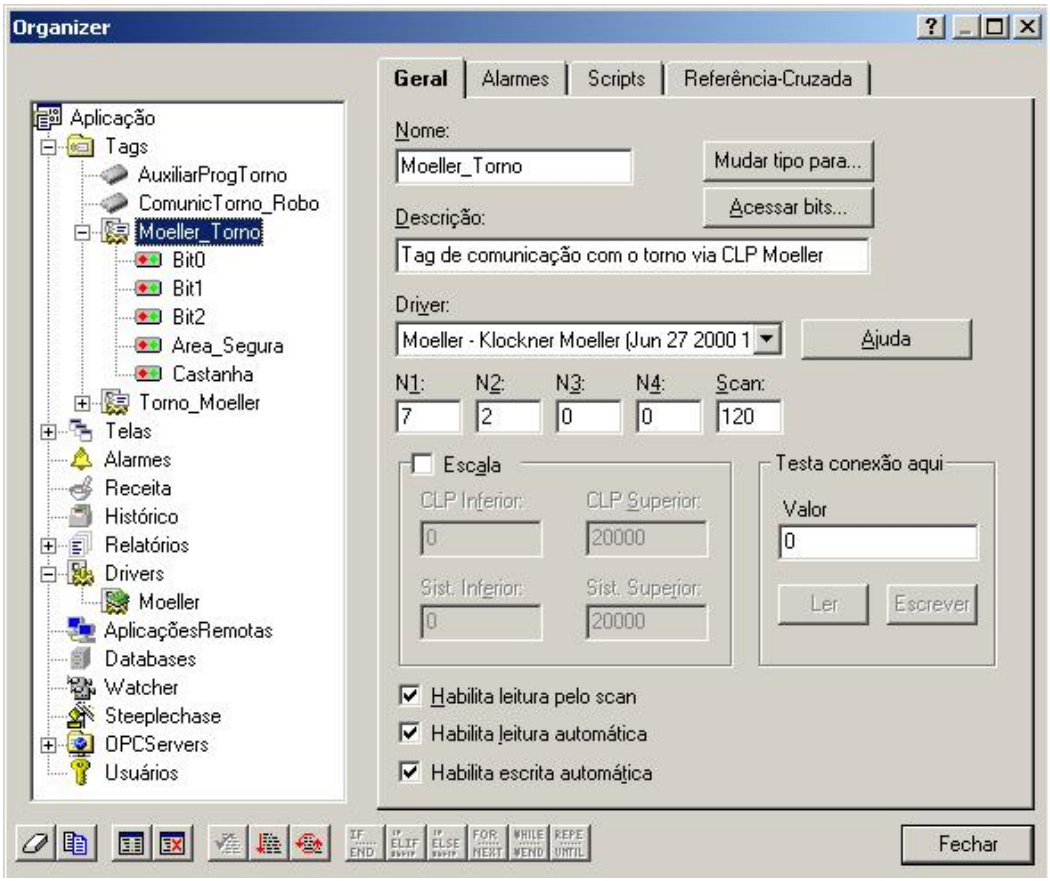


Figura 5.7. Parâmetros “N” da configuração do CLP Klockner Moeller.

Com a ajuda do manual do *driver*, fornecido pela Elipse, configuraram-se os seguintes parâmetros para o CLP Moeller:

Tabela 5.7. Configuração dos Parâmetros “N”

Parâmetro	Significado
N1 = 7	Tipo do PLC, n1=7 PS4-200
N2 = 2	Tipo de leitura, n2=2 *Merker/Word*
N3 = 0	*Variable address*
N4 = 0	*bit position in the word*

A parametrização adotada é idêntica para as duas *tags* CLP. Para que o CLP Moeller possa decodificar as informações recebidas do gerenciador e enviá-las ao torno, e vice-versa, foi necessário escrever um programa utilizando o *software* de programação Sucosoft S40, versão 5.0. O programa para o CLP Moeller pode ser visto na figura 5.8. Todos os programas desenvolvidos para integração da célula também se encontram no Anexo em CD-ROM.

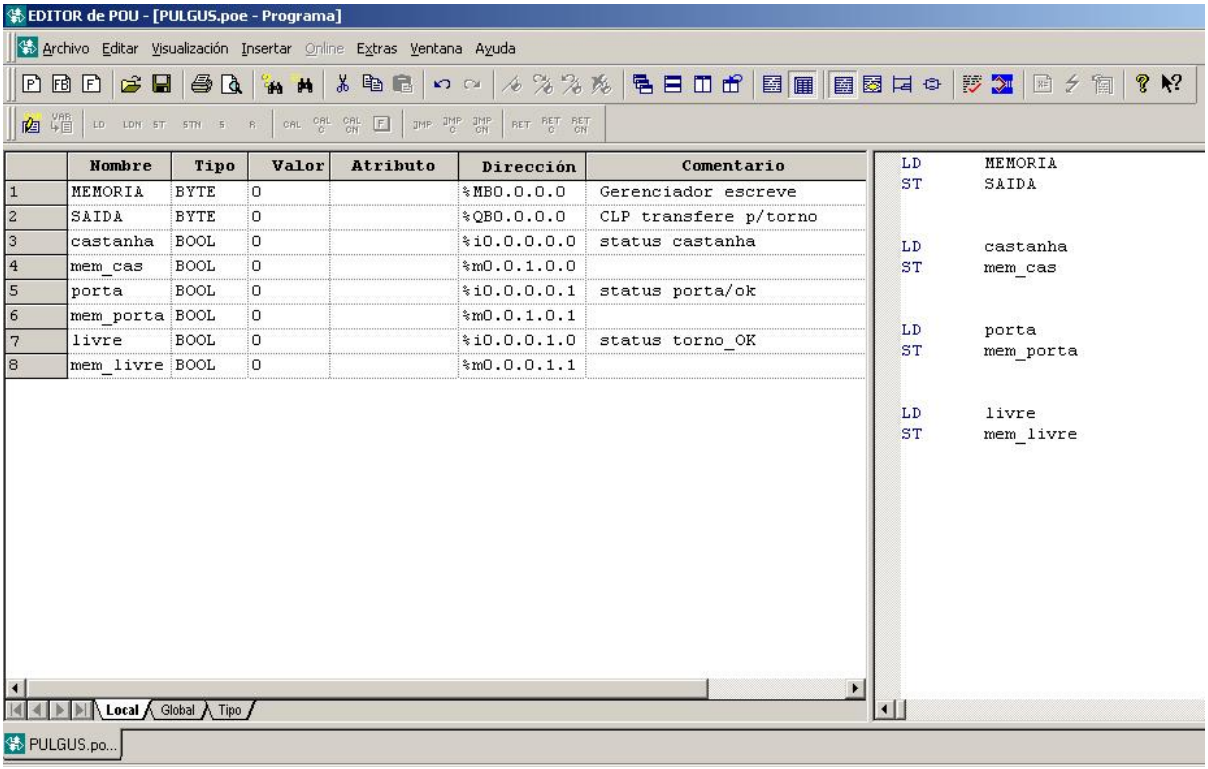


Figura 5.8. Programa do CLP Moeller do torno.

O programa inicialmente declara as variáveis, os endereços e o tipo de variável envolvida. Comentários foram inseridos para documentar o programa. Assim, é possível observar que as variáveis “MEMORIA” e “SAIDA” são do tipo “*BYTE*”, assumem valor inicial igual a zero e ocupam os endereços “MB0.0.0.0” e “QB0.0.0.0”, respectivamente. Na seqüência são definidas as variáveis do tipo “BOOL”, todas com valor inicial igual a zero: castanha (i0.0.0.0.0), mem_cas (m0.0.1.0.0), porta (i0.0.0.0.1), mem_porta (m0.0.1.0.1), livre (i0.0.0.1.0) e mem_livre (m0.0.0.1.1).

O programa encarregado de transferir os dados do gerenciador para o torno ocupa as duas primeiras linhas. A primeira carrega o valor da variável “MEMORIA” através do comando “LD” (*Load*), e na linha seguinte transfere este valor para a variável “SAIDA”, através do comando “ST” (*Storage*).

O programa que transfere dados do torno para o gerenciador é similar ao anterior, com a diferença de que são três as variáveis lidas do torno (castanha, porta e livre). Após carregar o conteúdo das variáveis, estas são transferidas para saídas (mem_cas, mem_porta e mem_livre). A figura 5.9 exemplifica o programa do CLP Moeller, apresentando também as conexões entre o CLP e o torno, que serão abordadas com mais detalhes no capítulo 5.2.3.

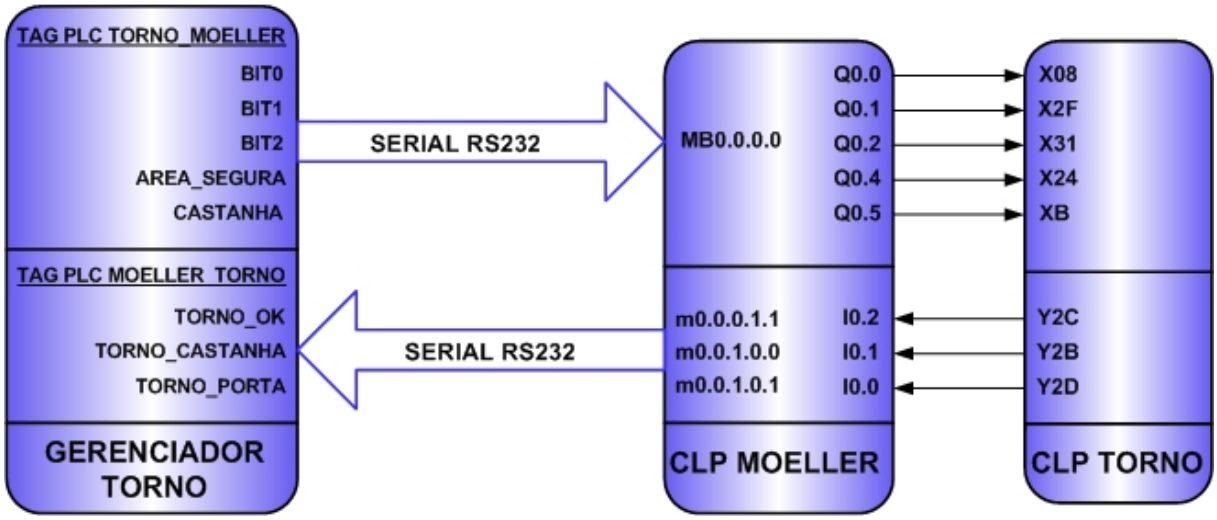


Figura 5.9. Esquema de conexões do CLP Moeller.

5.2.3 Integração entre CLP Moeller e o torno

A última etapa da integração é a conexão entre o CLP Moeller e o torno. O torno CNC Feeler, modelo FTC10, originalmente não está preparado para trocar dados com nenhum outro equipamento. Para realizar esta tarefa foram introduzidas mudanças no *hardware* e na programação do seu CLP, um Mitsubishi Meldas 50M. Estas modificações foram realizadas em 2004, por dois formandos do curso de graduação do IST (Instituto Superior Tupy), no âmbito do desenvolvimento do trabalho de conclusão de curso (HAAS e DETTRUZ, 2004). Para que seja possível acompanhar o complemento da integração do CLP Moeller com o torno, será apresentado um breve resumo das modificações introduzidas no torno. Todos os diagramas elétricos destas modificações se encontram no Apêndice B desta dissertação.

O passo inicial foi automatizar a porta do torno, que a princípio era acionada manualmente pelo operador. Para tanto foi instalado um cilindro pneumático com suas respectivas válvulas e um banco auxiliar de relés. Todos os sinais trocados entre o CLP Moeller e o CLP Meldas são eletricamente isolados. Isto se faz necessário devido ao fato de que existem diferenças de potenciais elétricos nas saídas dos CLPs. São utilizados dois bancos de relés auxiliares para efetuar o isolamento elétrico dos sinais.

Figura 5.10. Cilindro pneumático, porta do torno e banco de relés auxiliares (no detalhe).

A atuação das válvulas do cilindro pneumático é realizada por meio dos contatos RL2 (fechar porta) e RL3 (abrir porta). Os contatos do relé são acionados pelas saídas do CLP Meldas Y27 (abrir porta) e Y29 (fechar porta). As funções de abrir e fechar a porta já estavam previstas nas funções auxiliares M (*miscellaneous*) do torno, como M17 e M18 respectivamente, porém não estavam implementados na lógica do CLP Meldas. Para implementação da lógica utilizaram-se as memórias M617, para a saída Y27, e M618, para a saída Y29. A Figura 5.11 exemplifica estas implementações.

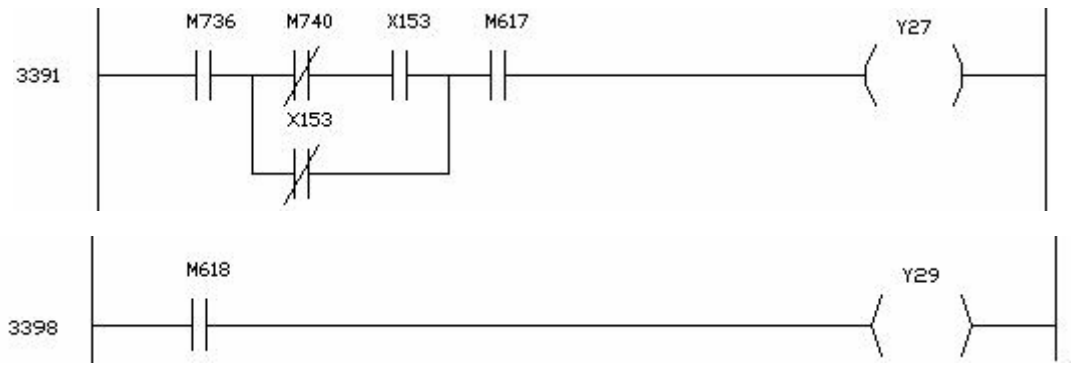


Figura 5.11. *Ladder* fechar e abrir porta.
(Fonte: HAAS e DETTRUZ, 2004)

O estado do torno é informado através do sinal TORNO OK/OUT. Para implementar este sinal foi necessário utilizar a saída Y2C do CLP Meldas, ligado ao relé auxiliar RL6. Quando o torno carrega positivamente um programa e inicia o processo de usinagem, a saída apresenta um nível zero (Y2C=0). Quando o torno estiver livre, ou seja sem programa carregado, com a porta e a castanha aberta, a saída apresentará um nível alto (Y2C=1), indicando que o torno está apto a receber uma solicitação de usinagem. O sinal de que o torno está apto a receber a carga de um novo programa exigiu, por questões de segurança, uma redundância no seu intertravamento. Somente com a indicação de que a porta está aberta, fornecida pela entrada interna X38=1 e que a castanha também está aberta (Y2B=0), é que a saída Y2C pode se acionada, indicando que o torno está livre conforme mostra o *Ladder* da figura 5.12.

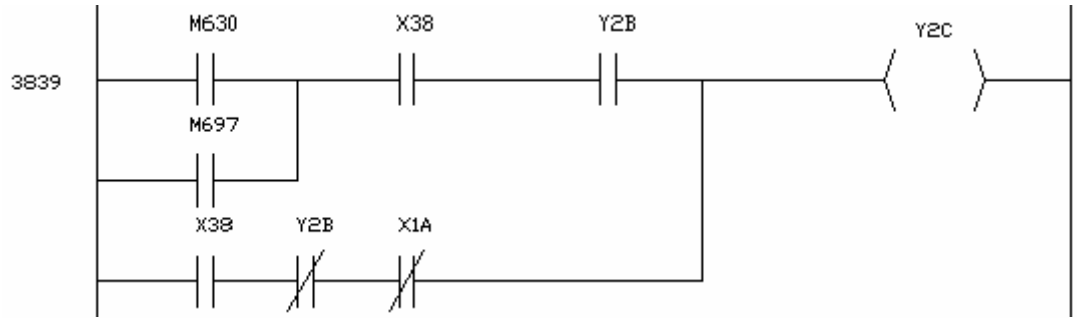


Figura 5.12. *Ladder* do torno livre/usinando.
(Fonte: HAAS e DETTRUZ, 2004)

A abertura e fechamento da castanha devem ser sincronizados com a movimentação do robô. Quando o robô carrega uma peça para usinagem, ele indica que a peça está na posição através da entrada XB do CLP Meldas. Este sinal foi colocado em paralelo com a chave S1/H2, que é o pedal onde o operador efetua esta operação de forma manual. A lógica de abertura e fechamento da castanha utiliza apenas um pulso para a mudança do seu estado inicial. O primeiro pulso abre, e o segundo fecha a castanha. O relé RL5 indica a condição da castanha através da saída Y2B do CLP Meldas, que está conectado à entrada I0.1 do CLP Moeller. A figura 5.13 mostra o *Ladder* de indicação de castanha aberta ou fechada. A memória M713 é utilizada pela lógica da castanha e indica que a mesma está fechada, sempre que sua saída apresente nível alto.

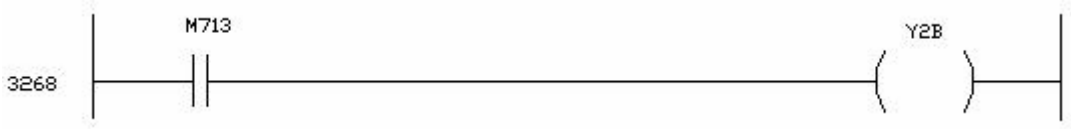


Figura 5.13. *Ladder* da castanha aberta/fechada.
(Fonte: HAAS e DETTRUZ, 2004)

Após a carga da peça a ser usinada, o torno aguarda a informação de que o robô se encontra em área segura, sinal proveniente da saída Q0.4 do CLP Moeller e entrada X24 do CLP Meldas. Somente após receber esta confirmação é que o robô fecha a porta e inicia a usinagem. O torno utiliza a saída X11 para indicar a partida (*start*) do programa de usinagem, no modo automático. Para indicar que o torno pode iniciar o procedimento de usinagem, quando o robô estiver em área segura, colocou-se um contato da entrada X24 em paralelo com a entrada X11, atuando a memória M201, que é o sinal de *start* para o restante da lógica.

Figura 5.14. *Ladder* de robô em área segura e *start*.
(Fonte: HAAS e DETTRUZ, 2004)

A escolha do programa de usinagem, que deverá estar previamente carregado no torno, é realizada pela combinação BCD (*Binary Coded Decimal*) de três *bits* proveniente do CLP Moeller. Estes *bits* estão conectados nas entradas X08, X2F e X31 do CLP Meldas. A tabela 5.8 mostra a relação entre a *tag* ProgUsinagem, a *tag* AuxiliarProgTorno, a combinação BCD na saída do CLP Moeller, o programa correspondente acionado no torno e a peça fabricada.

Tabela 5.8. Relação entre *tags*, sinais do CLP Moeller e programas.

ProgUsinagem	AuxiliarProgTorno	BCD	Programa	Peça fabricada
" "	0	000	Nenhum	Nenhuma
PBR	1	001	Prog 01	Peão Branco
PPR	2	010	Prog 01	Peão Preto
BBR	3	011	Prog 04	Bispo Branco
BPR	4	100	Prog 04	Bispo Preto
DBR	5	101	Prog 05	Dama Branca
DPR	6	110	Prog 05	Dama Preta

O programa para usinagem de um peão branco ou um peão preto é exatamente o mesmo. Esta lógica é válida também para o bispo e a dama, que completam a família de peças direcionadas para fabricação no torno. A diferença está na matéria-prima empregada e na mesa do berço de usinagem, utilizada pelo armazém no fornecimento de matéria-prima. No caso das peças denominadas "brancas" utiliza-se o alumínio como matéria-prima. Já no caso das peças "pretas" pode-se utilizar latão ou cobre. De qualquer forma, a escolha de diferentes cores para a mesma peça foi empregada somente como um recurso para comprovar a flexibilidade oferecida pelo sistema no que se refere ao número de diferentes peças que podem ser processadas pelo torno. Para aumentar o número de combinações de programas basta aumentar o número de *bits* na combinação BCD. A limitação neste caso se dará pelo número de portas de saída disponíveis no CLP, ou pela capacidade de memória para programas do torno.

As entradas X08, X2F e X31 do CLP Meldas estão ligadas, via programação *Ladder*, a posições de memórias auxiliares do torno. Estas memórias estão inseridas dentro do programa principal (mestre) de usinagem do torno, associadas a instruções IF e GOTO. Todos os programas de usinagem, que são previamente carregados, fazem parte de um único programa principal. A chamada de cada programa ocorre de forma semelhante à chamada de uma sub-rotina, disparada pelo desvio condicional da instrução IF. Se a condição for verdadeira, o programa executa a instrução GOTO, indo até a linha onde inicia o programa desejado. Ao fim da usinagem o programa retorna ao corpo principal do programa mestre, aguardando a próxima instrução válida, dada pela ativação das memórias ligadas à combinação BCD recebida do CLP Moeller.

5.2.4 Integração entre gerenciador, CLP Moeller e torno

A integração entre o gerenciador do torno, o CLP Moeller e o torno possibilita que este conjunto se integre como um recurso produtivo da FMC. As figuras 5.15 e 5.16 mostram respectivamente: um resumo da arquitetura de troca de informações entre os componentes analisados, e o CLP Moeller, banco auxiliar de relés e conexões físicas.

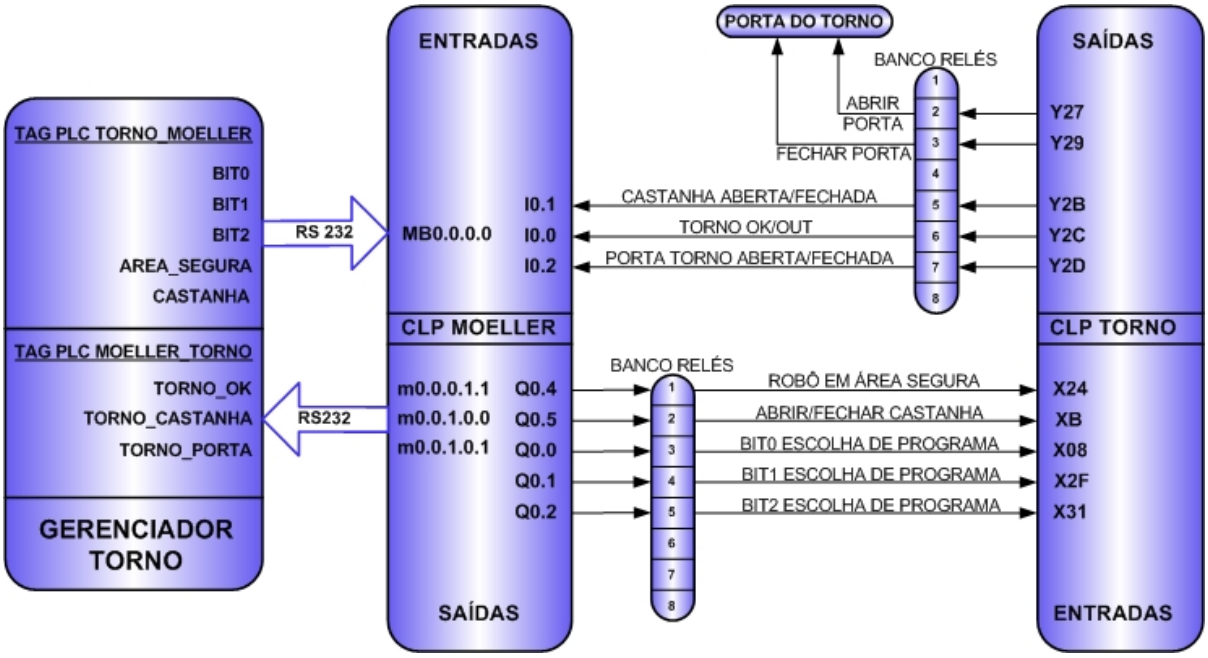


Figura 5.15. Arquitetura da troca de informações entre gerenciador, CLP Moeller e torno.

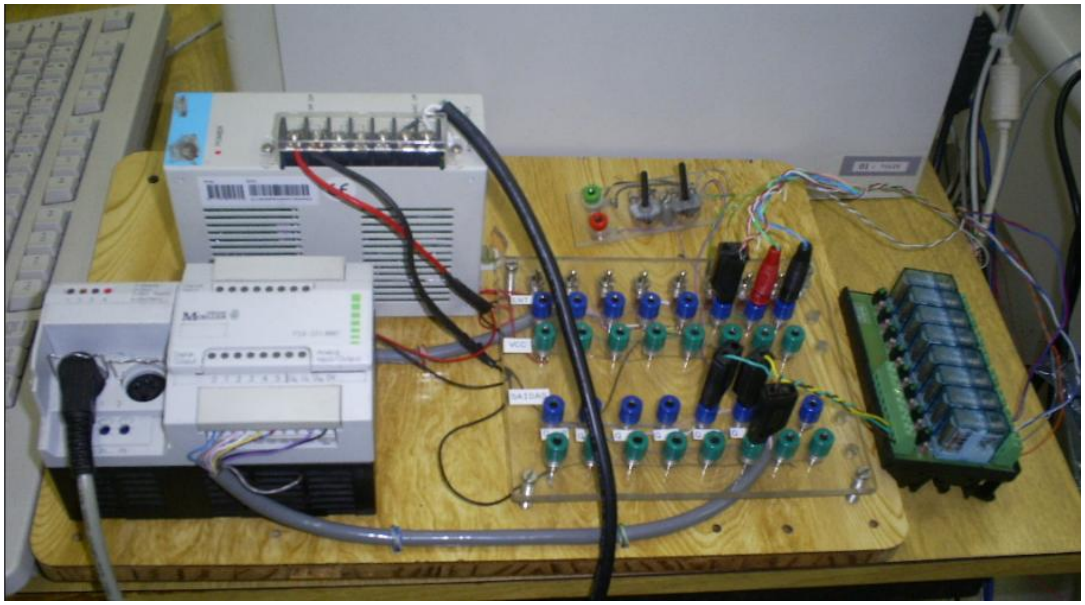


Figura 5.16. CLP Moeller, banco auxiliar de relés e conexões.

5.3 GERENCIADOR DA FRESADORA E GERENCIADOR DO CNC R1

A construção dos gerenciadores do Centro de Usinagem Feeler, modelo FV600, e do CNC R1, seguiu todos os passos adotados para o torno. As interfaces foram definidas, os aplicativos foram desenvolvidos no Elipse SCADA e os programas do CLP também foram escritos. A fresadora não pode efetivamente ser incorporada à célula porque não possui um sistema de fixação automático, que

poderia ser uma morsa hidráulica. Sem este dispositivo não é possível a operação autônoma do CNC, pois exigiria um operador para fixar a peça a ser usinada, o que também invalida a ação do robô, pois a *priori* é ele o responsável pelo carregamento e descarregamento do CNC. Independente da limitação imposta pela falta de um dispositivo de fixação automático, todo o desenvolvimento do gerenciador será apresentado de forma resumida, pois possui grande similaridade com o gerenciador do torno.

No caso do CNC R1, não existe de fato nenhuma máquina. Ela será simulada para testar as funcionalidades do sistema e para demonstrar a modularidade dos gerenciadores. O formato destes gerenciadores é semelhante para todos os equipamentos CNC, pois está baseado em um conjunto de interfaces mínimas que atendem às especificações de um grande número de CNCs. Outra razão para incluir esta máquina "virtual" na célula deve-se ao fato de que o espaço de trabalho do robô poderia, hipoteticamente, atender a uma terceira máquina CNC, que pode no futuro vir a ser efetivamente instalada.

Partindo da definição das interfaces mínimas para a fresadora e para o CNC R1, e convertendo-as para *tags* a serem utilizadas no Elipse SCADA, temos:

Tabela 5.9. Conversão de interfaces da fresadora e do R1 para *tags*.

INTERFACES	*TAG* PARA FRESA	*TAG* PARA R1
PROGRAMA A EXECUTAR	ProgUsinagem	ProgUsinagem
ROBÔ LIVRE/EM MOVIMENTO	ROBO_PHOME	ROBO_PHOME
ROBÔ EM CARGA/DESCARGA DE PEÇA	ROBO_PECA_NA_MORSA	ROBO_PECA_NO_R1
ROBÔ EM ÁREA SEGURA/DENTRO DO CNC	ROBO_AREA_SEGURA	ROBO_AREA_SEGURA
CNC LIVRE/USINANDO	FRESA_OK	R1_OK
PORTA ABERTA/FECHADA	FRESA_PORTA	R1_PORTA
DISPOSITIVO DE FIXAÇÃO ABERTO/FECHADO	FRESA_MORSA	R1_MORSA

O significado das *tags* que serão utilizados para construção dos gerenciadores da fresadora e do CNC R1 está detalhado na tabela 5.10.

Tabela 5.10. Significado das *tags* para os gerenciadores da fresadora e R1

TAG	VALOR=0	VALOR=1
ProgUsinagem	Nenhum	Nenhum
ROBO_PHOME	Em movimento	Em Phome (Robô está livre)
ROBO_PECA_NA_MORSA	Peça fora da morsa	Peça está na morsa
ROBO_PECA_NO_R1	Peça fora do disp. de fix. de R1	Peça está no disp. de fix. de R1
ROBO_EM_AREA_SEGURA	Robô em área de risco	Robô está em área segura
FRESA_OK e R1_OK	CNC está usinando (OUT)	CNC está livre (OK)
FRESA_PORTA e R1_PORTA	Porta do CNC está fechada	Porta do CNC está aberta
FRESA_MORSA	Morsa da fresadora está aberta	Morsa da fresa está fechada
R1_DISP_FIX	Disp. de fix. de R1 está aberto	Disp. de fix de R1 está fechado

Um resumo das *tags* utilizadas para a construção do gerenciador do centro de usinagem (fresadora) e do gerenciador CNC R1, pode ser visto na Figura 5.17.

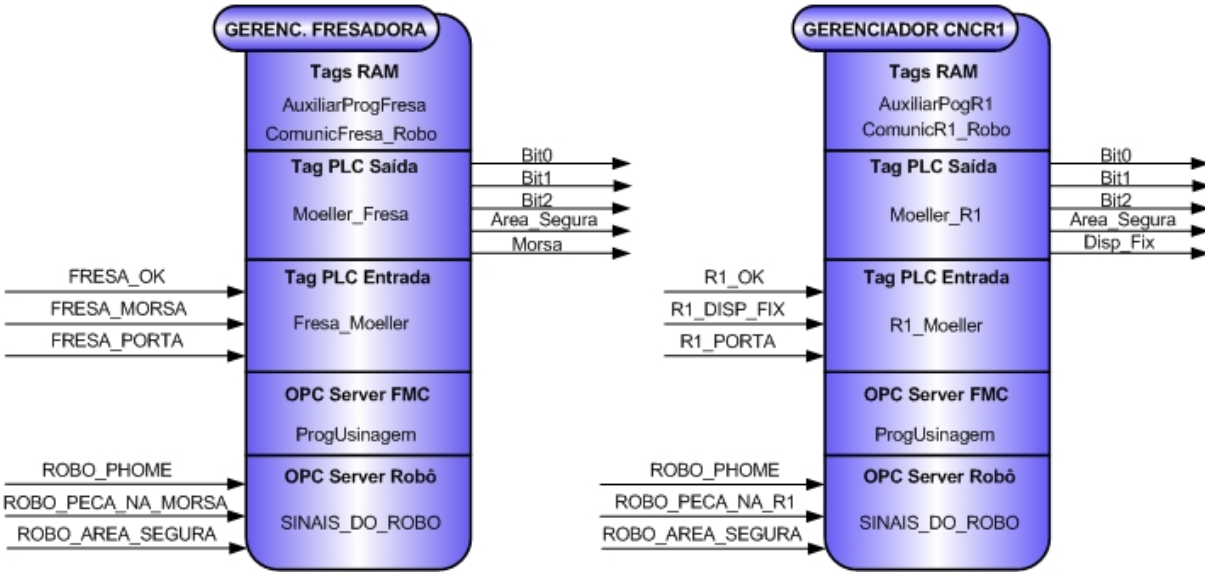


Figura 5.17. *Tags* utilizadas pelos gerenciadores da fresadora e do CNC R1.

É possível observar na estrutura de *tags*, e no significado dos mesmos, que existe um paralelismo entre os gerenciadores dos equipamentos CNC. Isto não é coincidência, mas resultado da padronização inicial de um conjunto de interfaces para os recursos produtivos da célula. Se houvesse mais uma máquina CNC, independente de ser um torno ou um centro de usinagem, esta teria exatamente a mesma estrutura, mudando apenas o nome e o dispositivo de fixação (morsa, castanha, pinça, etc.). Desta forma, a construção dos gerenciadores no supervisório Elipse SCADA, os *scripts*, *driver*s, configurações, programas de CLP e as conexões entre os gerenciadores, CLPs e CNCs é padronizada e adota a mesma estrutura analisada em detalhes na construção do gerenciador do torno. Em razão desta padronização, ou modularidade, somente serão analisados na seqüência deste trabalho aqueles itens que diferem em alguma particularidade do gerenciador do torno.

A relação entre a *tag* ProgUsinagem e as *tags* AuxiliarProgFresa e AuxiliarProgR1, e o significado de cada uma dela é mostrada na tabela 5.11.

Tabela 5.11. Relação entre *tags* e ações correspondentes

ProgUsinagem	AuxiliarProgTorno	AuxiliarProgR1	Significado
“ “	0	0	AGUARDANDO INSTRUÇÃO
TBR	1		FRESADORA USINA TORRE BRANCA
TPR	2		FRESADORA USINA TORRE PRETA
CBR	3		FRESADORA USINA CAVALO BRANCO
CPR	4		FRESADORA USINA CAVALO PRETO
RBR	5		FRESADORA USINA REI BRANCO
RPR	6		FRESADORA USINA REI PRETO
M1R1		1	CNC R1 RECEBE DA MESA1
M2R1		2	CNC R1 RECEBE DA MESA2
EMERG	15	15	EMERGÊNCIA ACIONADA

A interface com o usuário, neste caso do gerenciador da fresadora, assim como algumas das *tags* do *organizer*, é mostrada pela figura 5.18.

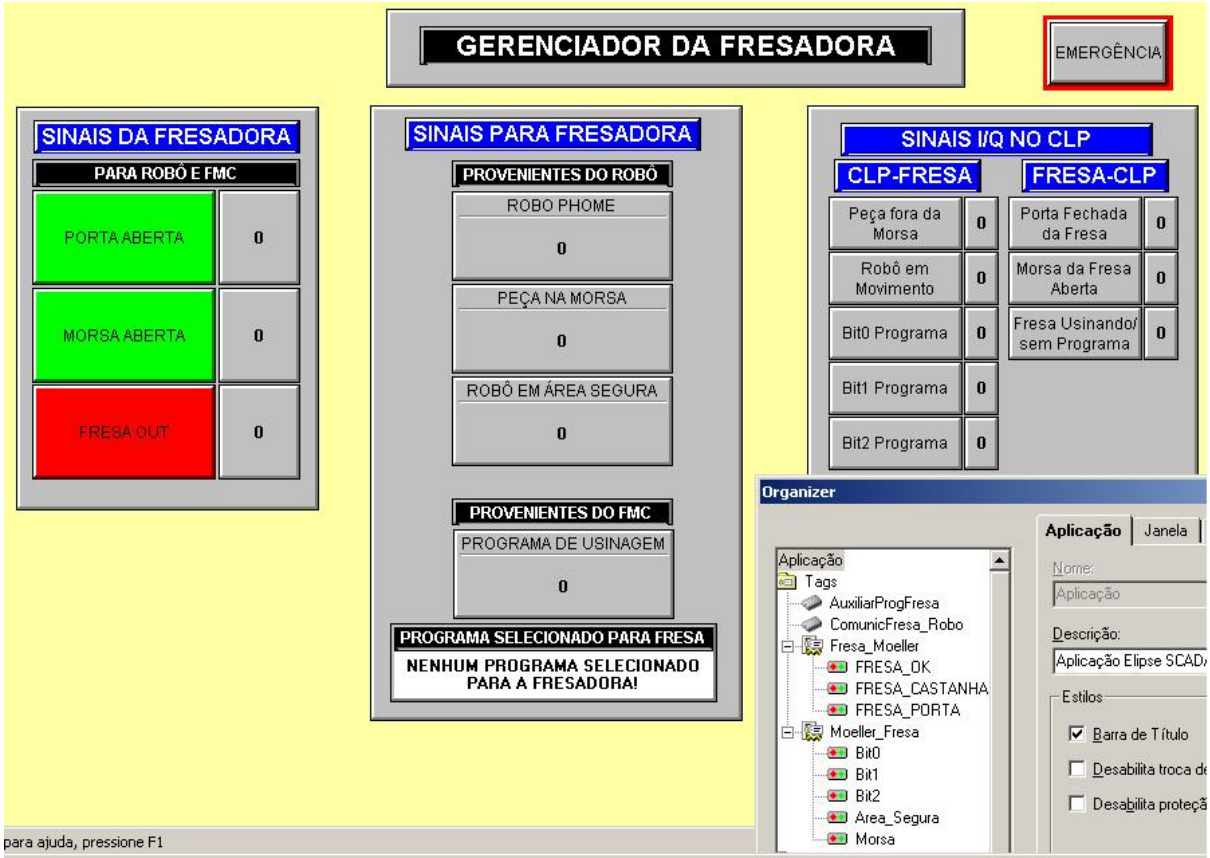


Figura 5.18. Interface com o operador do gerenciador da fresadora e *tags* do *organizer*.

5.4 GERENCIADOR DO ROBÔ

O robô é o responsável pelo abastecimento de matéria-prima para os equipamentos CNC e o retorno das peças usinadas para o armazém automático. No item 4.2.2 definiram-se as interfaces mínimas para o robô. As mesmas são apresentadas pela tabela 5.12 com a respectiva conversão para as *tags* que serão utilizadas na construção do gerenciador do robô.

Tabela 5.12. Conversão de interfaces do robô para *tags*.

INTERFACES	*TAG*
ROBÔ LIVRE/EM MOVIMENTO	ROBO_PHOME
ROBÔ NA CASTANHA DO TORNO	ROBO_PECA_NA_CASTANHA
ROBÔ NA MORSA DA FRESADORA	ROBO_PECA_NA_MORSA
ROBÔ NO DISP. DE FIX. DO CNC R1	ROBO_PECA_NA_R1
ROBÔ NA MESA1 DO ARMAZÉM	ROBO_M1
ROBÔ NA MESA2 DO ARMAZÉM	ROBO_M2
ROBÔ EM ÁREA SEGURA	ROBO_AREA_SEGURA
PROGRAMA A SER EXECUTADO	ProgUsinagem
PRESENÇA DE *PALLET* NA MESA1	ASAR_M1
PRESENÇA DE *PALLET* NA MESA2	ASAR_M2
DISP. FIX. DE R1 ABERTO/FECHADO	R1_DISP_FIX
PORTA DE R1 ABERTA/FECHADA	R1_PORTA
CNC R1 LIVRE/USINANDO	R1_OK
MORSA DA FRESADORA ABERTA/FECHADA	FRESA_MORSA
PORTA DA FREADORA ABERTA/FECHADA	FRESA_PORTA
FRESADORA LIVRE/USINANDO	FRESA_OK
CASTANHA DO TORNO ABERTA/FECHADA	TORNO_CASTANHA
PORTA DA TORNO ABERTA/FECHADA	TORNO_PORTA
TORNO LIVRE/USINANDO	TORNO_OK

O gerenciador do robô necessita da informação dos estados dos equipamentos CNC (livres/ocupados, porta aberta/fechada e dispositivo de fixação aberto/fechado) para estabelecer os sincronismos das ações de carregamento e descarregamento. Ao mesmo tempo também informa aos gerenciadores dos CNCs seus próprios estados (coloca/retira peça no dispositivo de fixação dos CNCs, e se está em área segura ou em movimentação com risco de colisão). O gerenciador do armazém informa da chegada de *pallet* com carga na mesa1 ou mesa2 dos berços de usinagem, e é informado sobre a devolução das peças já usinadas nas mesmas mesas, por parte do gerenciador do robô. Com o gerenciador da FMC, a troca de mensagens se resume à indicação de que o robô está livre ou usinando e ao recebimento do programa de usinagem para execução.

O conjunto de *tags* utilizados pelo gerenciador do robô está representado pela figura 5.19.

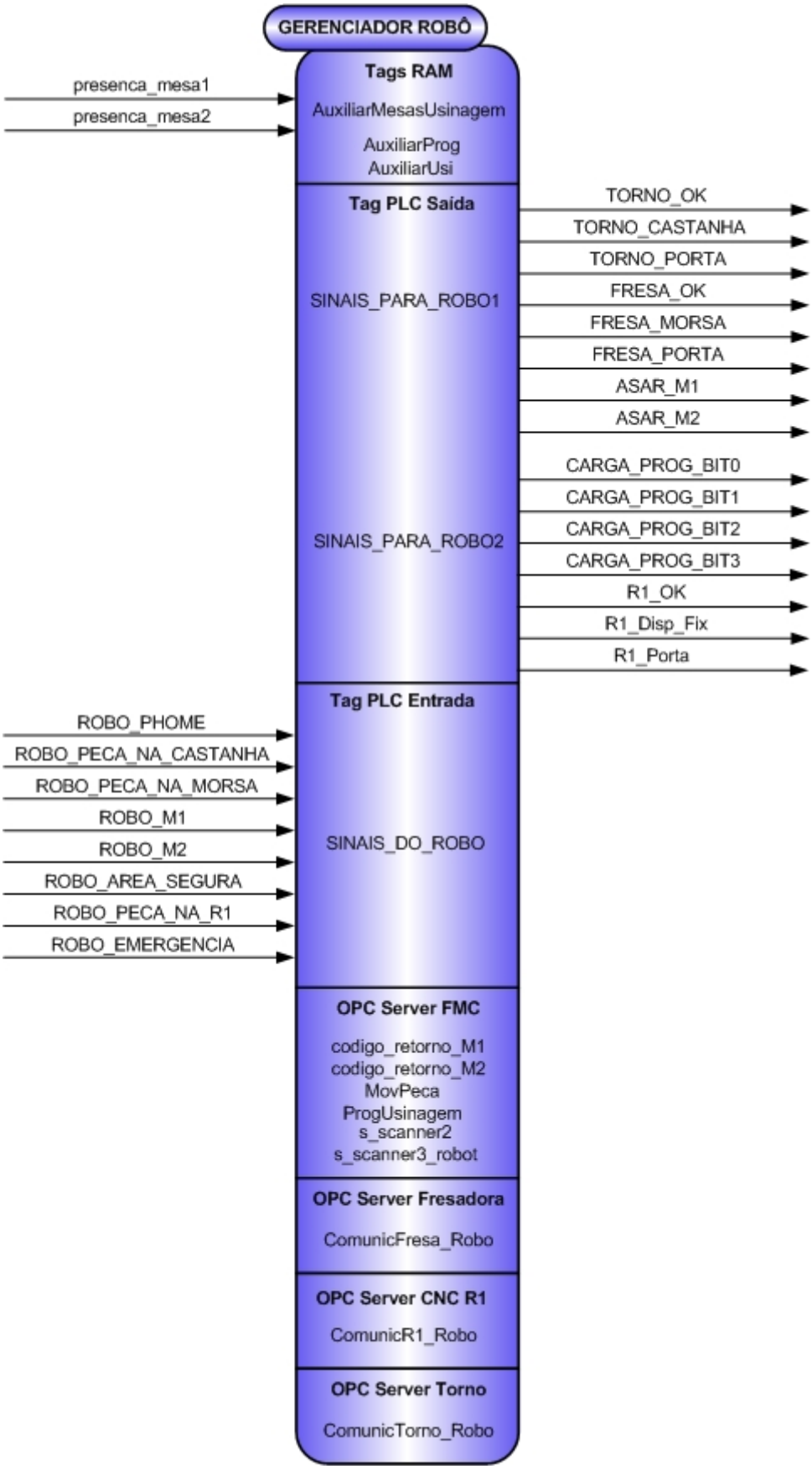


Figura 5.19. Conjunto de *tags* utilizado pelo gerenciador do robô.

As informações são recebidas dos outros gerenciadores por meio de *tags* OPC. Neste caso o aplicativo do gerenciador do robô comporta-se como cliente OPC, enquanto os demais gerenciadores assumem o papel de servidores OPC. Observa-se que não existe na configuração das *tags* um servidor OPC do armazém automático. Isto se deve ao fato de que o aplicativo gerenciador do armazém e o gerenciador FMC foram desenvolvidos em um único aplicativo, por razões que são abordadas no item 5.5. Desta forma, tanto as informações provenientes do armazém como do gerenciador FMC, são importadas do servidor OPC FMC.

O gerenciador do robô utiliza três *tags* do tipo RAM como auxiliares. A *tag* AuxiliarMesasUsinagem, que é desmembrada nas *tags* presença_mesa1 e presença_mesa2, indica a presença de *pallet* na mesa1 ou mesa2 dos berços de usinagem. A informação original está na *tag* s_scanner2, e é desmembrada através do uso de *script* de programação (todos os *scripts* estão detalhados no Apêndice D). As *tags* AuxiliarProg e AuxiliarUsi tem a função de alimentar objetos do tipo texto na interface do operador, para indicar qual programa de usinagem está sendo solicitado e qual programa de movimentação o robô está executando. Estas *tags* também são atualizadas por *script*, diretamente da *tag* ProgUsinagem.

Para comunicação do gerenciador com o CLP são utilizados dois conjuntos de *tags* tipo PLC. O primeiro conjunto corresponde à saída de informações do gerenciador para o CLP e é composto pelas *tags* SINAIS_PARA_ROBO1 e SINAIS_PARA_ROBO2. O segundo conjunto recebe informações do CLP e é representada pela *tag* SINAIS_DO_ROBO.

5.4.1 Aplicativo gerenciador do robô no Elipse SCADA

Definida a arquitetura e as *tags* necessárias, inicia-se a construção do aplicativo gerenciador do robô no *software* Elipse SCADA. A figura 5.20 apresenta a interface do gerenciador com o operador.

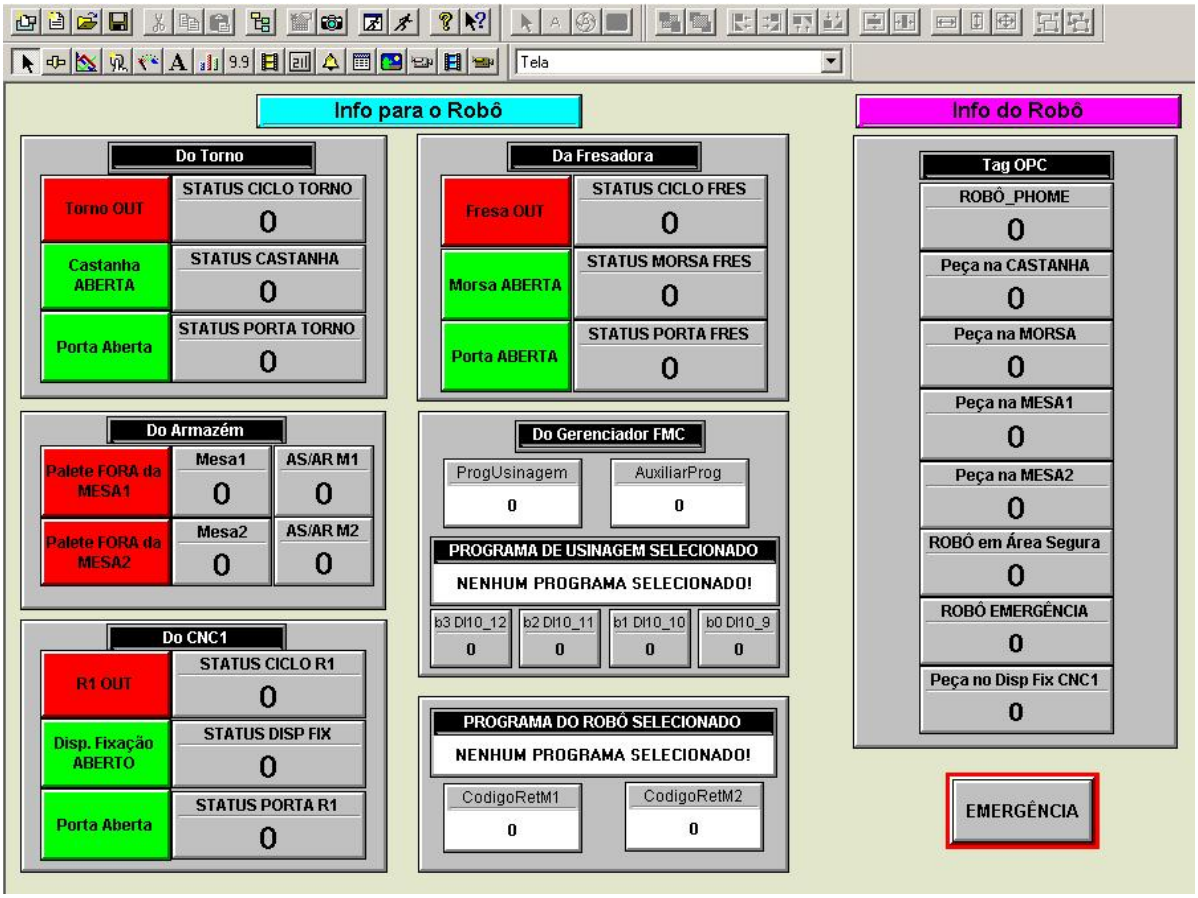


Figura 5.20. Interface do gerenciador do robô com operador.

A configuração das *tags* OPC foi realizada de forma semelhante ao gerenciador do torno. O gerenciador do robô funciona como cliente OPC, e os demais gerenciadores que fornecem informações ao robô, atuam como servidores OPC. A figura 5.21 mostra a estrutura das *tags* OPC, importadas dos outros gerenciadores assim como um detalhe parcial do *script*, disparado pelo evento OnValueChanged da *tag* ProgUsinagem, que atualiza as *tags* tipo Ram AuxiliarUsi e AuxiliarProg.

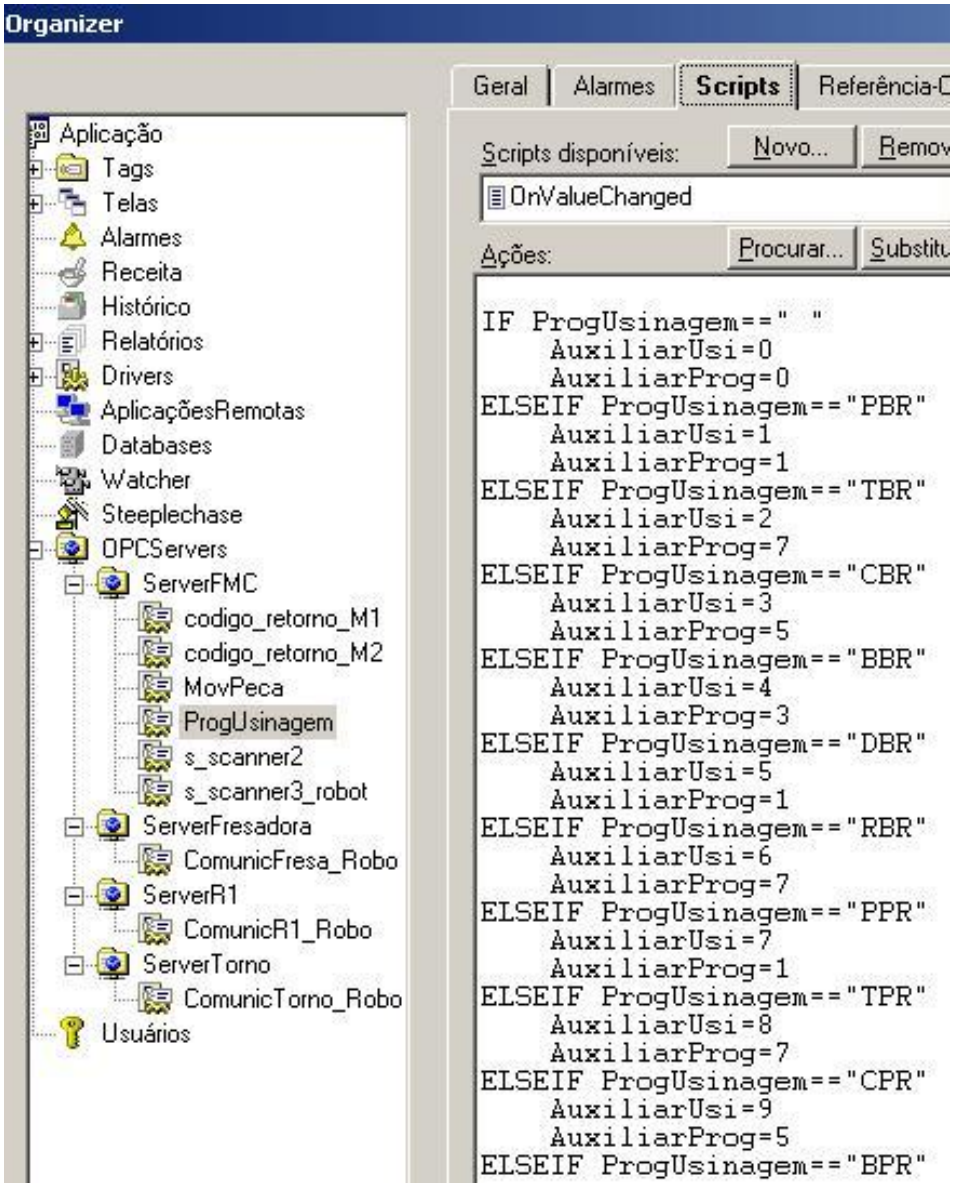


Figura 5.21. Estrutura das *tags* OPC e detalhe de um *script*.

A *tag* ProgUsinagem contém a informação do código da peça que o usuário solicita para fabricação. A partir dela são carregados os programas de usinagem nos equipamentos CNC, o programa que o robô deverá executar e o encaminhamento da matéria-prima correspondente do armazém para as mesas de usinagem. A tabela 5.13 complementa as informações parciais da figura anterior (5.21), referente ao *script* associado à *tag* ProgUsinagem, e também relaciona o código de usinagem solicitado, à mesa em que o armazém automático apresentará o *pallet* com a matéria-prima, o CNC que processará a peça e o programa do robô para carregamento dos equipamentos CNC.

Tabela 5.13. Relação entre *tags*, mesa de usinagem, CNC e programa do robô.

ProgUsinagem	AuxiliarUsi	AuxiliarProg	Mesa	CNC	Programa Robô
“ “	0	0			
PBR	1	1	1	Torno	M1T1
PPR	7	1	1	Torno	M1T1
TBR	2	7	2	Fresadora	M2F1
TPR	8	7	2	Fresadora	M2F1
CBR	3	5	1	Fresadora	M1F1
CPR	9	5	1	Fresadora	M1F1
BBR	4	3	2	Torno	M2T1
BPR	10	3	2	Torno	M2T1
DBR	5	1	1	Torno	M1T1
DPR	11	1	1	Torno	M1T1
RBR	6	7	2	Fresadora	M2F1
RPR	12	7	2	Fresadora	M2F1
M1R1	13	9	1	CNC R1	M1R1
M2R1	14	11	2	CNC R1	M2R1
EMERG	15	15			

As *tags* código_retorno_M1, código_retorno_M2, MovPeca e s_scanner3_robot, importadas do servidor OPC FMC não são essenciais ao gerenciador do robô, mas fornecem informações sobre estados intermediários que ajudaram no desenvolvimento do sistema. Como existe a possibilidade que a pesquisa tenha continuidade, optou-se pela permanência destas *tags* no aplicativo, para facilitar a monitoração de sinais e eventos importantes na eventual ampliação ou modificação do sistema.

Os estados dos equipamentos CNC são coletados através das *tags* OPC ComunicTorno_Robo, ComunicFresa_Robo e ComunicR1_Robo. Elas são desmembradas por meio de *script* (apresentados no Apêndice D) e atualizam as informações da condição de livre/ocupado, a condição da porta e do dispositivo de fixação de cada CNC.

A figura 5.22 apresenta o restante das *tags* utilizadas pelo gerenciador do robô, ressaltando no detalhe a parametrização (parâmetros “N”) do *driver* Prodave, associado à *tag* tipo PLC SINAIS_PARA_ROBO1, que envia informações do gerenciador para o controlador do robô, por meio de um CLP Siemens S7-300.

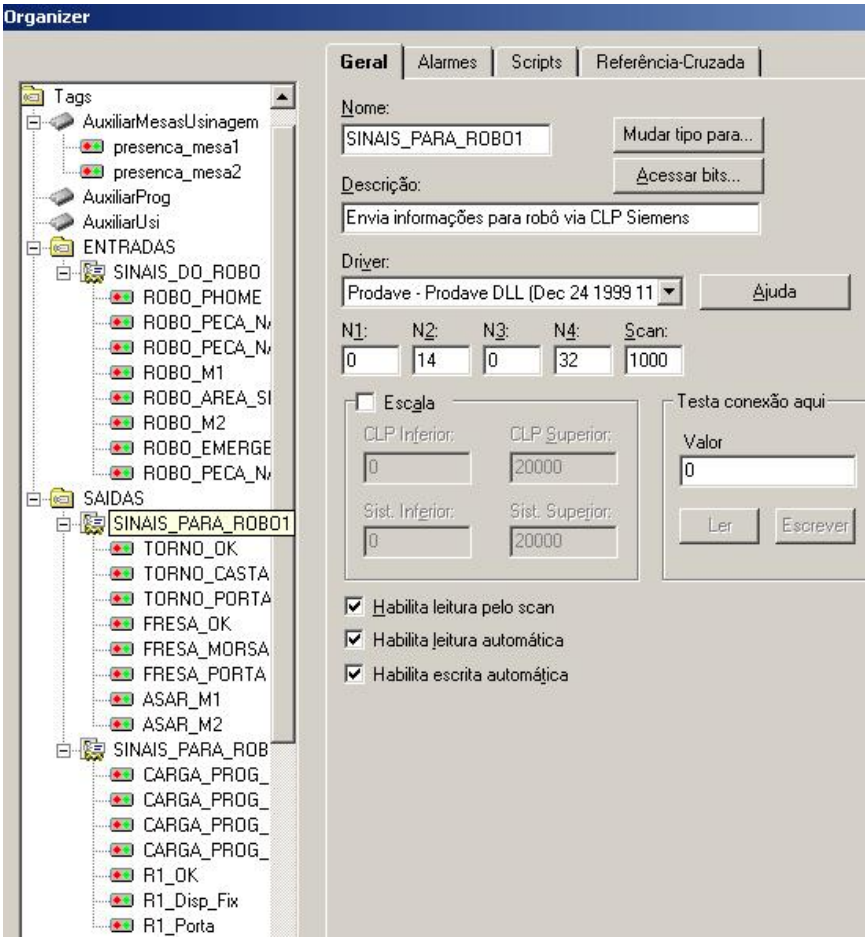


Figura 5.22. *Tags* utilizadas pelo gerenciador do robô e parametrização do *driver* Prodave.

5.4.2 Integração do gerenciador do robô com CLP Siemens

Para que o gerenciador do robô pudesse trocar informações com o controlador do robô, foi necessário utilizar um CLP, neste caso um Siemens, modelo S7-300. O gerenciador precisa receber do robô informações referentes a seus estados e posições. O gerenciador também necessita enviar ao robô as informações sobre estados dos equipamentos CNC, presença de *pallet* nas mesas de usinagem do armazém e sobre o programa de movimentação que o robô deve executar.

Para utilização do CLP Siemens S7-300, inicialmente é preciso instalar e configurar o *software* Prodave, fornecido pelo fabricante do equipamento. Posteriormente deve-se parametrizar o *driver* Prodave, fornecido pela Elipse. Na configuração do *software* Prodave versão 5.00, para os "Parâmetros de Estação" instalou-se uma interface padrão "MPI", endereço "1", "Timeout" de 30s e "Transmission rate" de 187,5 Kbps. Na janela de "Conexão Local", configurou-se a porta serial COM2, com velocidade de 19200 bps. Já na configuração do *driver* Prodave, os parâmetros "P" foram

mantidos com valores iguais a zero, desta forma a parametrização será realizada pelos parâmetros “N”, diretamente associados às *tags* tipo PLC do *software* Elipse SCADA.

A figura 5.22 mostra os parâmetros “N” configurados para a *tag* SINAIS_PARA_ROBO1: (N1=0, N2=14, N3=0, N4=32). A configuração da *tag* SINAIS_PARA_ROBO2: (N1=0, N2=14, N3=0, N4=33) e da *tag* SINAIS_DO_ROBO: (N1=0, N2=15, N3=0, N4=16). O significado destes parâmetros, segundo manual do *driver,* são os seguintes:

Tabela 5.14. Configuração dos Parâmetros “N” para o *driver* Prodave

Parâmetro (*Tag*)	Significado
N1 = 0	Número da Conexão (1)
N2=14, N3=0, N4=32	N2=14 R/W, *Outputs Bytes*
N2=14, N3=0, N4=33	N2=14 - R/W, *Outputs Bytes*
N2=15, N3=0, N4=16	N2=15 – *Read Only*, *Inputs Bytes*

Para que o CLP Siemens possa processar e transferir as informações recebidas do gerenciador para o torno, e vice-versa, desenvolveu-se um programa para o CLP utilizando o *software* Simatic Managers, versão 5.3V, da Siemens (figura 5.23).

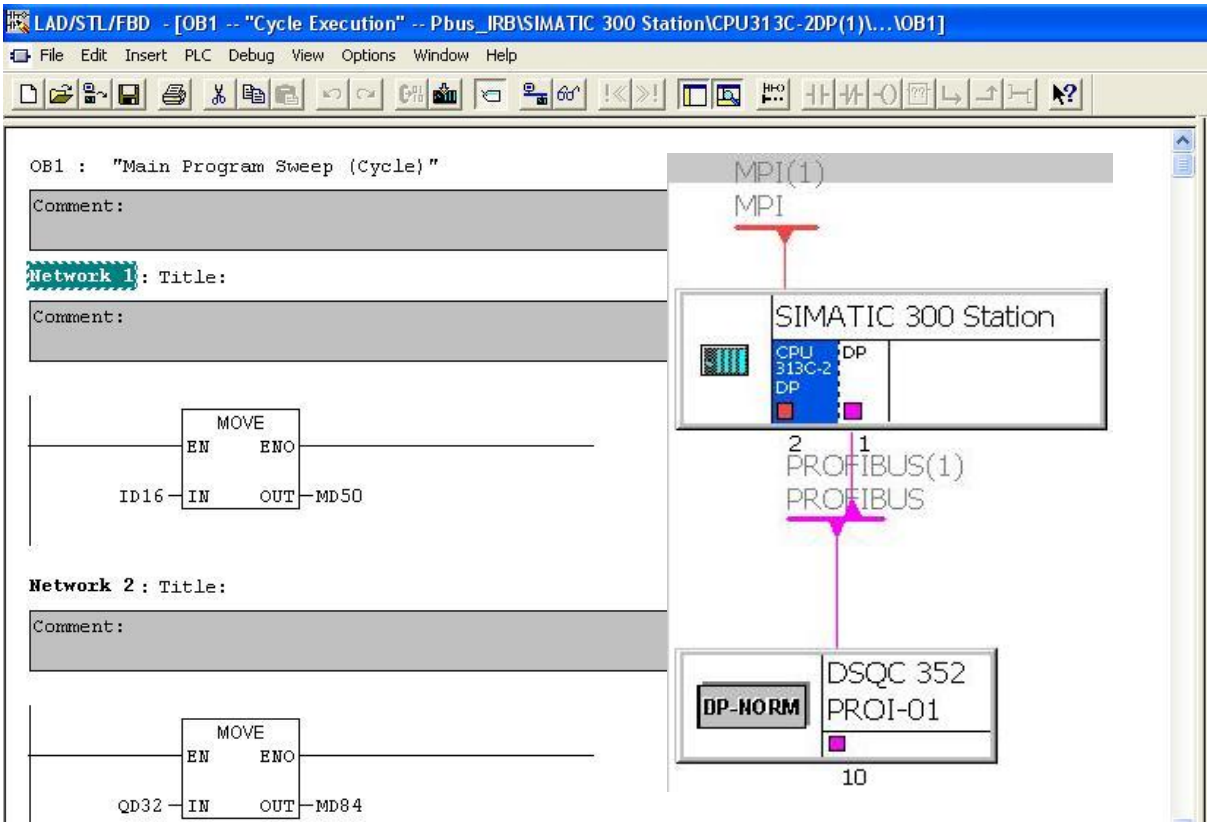


Figura 5.23. Programa do CLP Siemens e topologia da rede.
(Fonte: Autoria de Léo Schirmer, em: CHAVES e ALBANO, 2006)

O programa é composto por dois blocos e se baseia parcialmente em um programa desenvolvido anteriormente por dois formandos de graduação do IST (CHAVES e ALBANO, 2006). O primeiro bloco do programa recebe as informações provenientes do robô através da rede Profibus. Estas informações de entrada têm o formato de *bytes* e são recebidas sequencialmente a partir da porta ID16, conforme configurado no *driver* Prodave. Após receber as entradas, o programa carrega esta informação em uma memória (MD50) e a transfere para o aplicativo do gerenciador do robô, via rede MPI (*Message Passing Interface*). Há um conjunto de 16 IDs, o que totaliza um total de 128 *bits* disponíveis para entrada de dados no CLP. A *tag* SINAIS_DO_ROBO utiliza 8 *bits* de dados para comunicar os estados e posições do robô ao gerenciador, todos agrupados na ID16 do CLP Siemens.

O segundo bloco de programação trata da transferência dos dados do gerenciador para o controlador do robô. Duas *tags* de saída (SINAIS_PARA_ROBO1 e SINAIS_PARA_ROBO2) enviam seus dados via rede MPI para as portas QD32 e QD33. Estes dados são movidos para a memória MD84, que as transfere para o robô, utilizando a rede Profibus. Neste caso também estão disponíveis 16 conjuntos de QDs, com 128 *bits* de saída no total. As duas *tags* de saída utilizam 15 *bits* de dados para comunicação entre o gerenciador e o torno. A figura 5.24 exemplifica as conexões entre gerenciador, CLP Siemens e robô, detalhando a decomposição em *bits* das *tags* PLC e as correspondências destas com as entradas e saídas digitais do controlador do robô, que serão aprofundadas no capítulo 5.4.3.

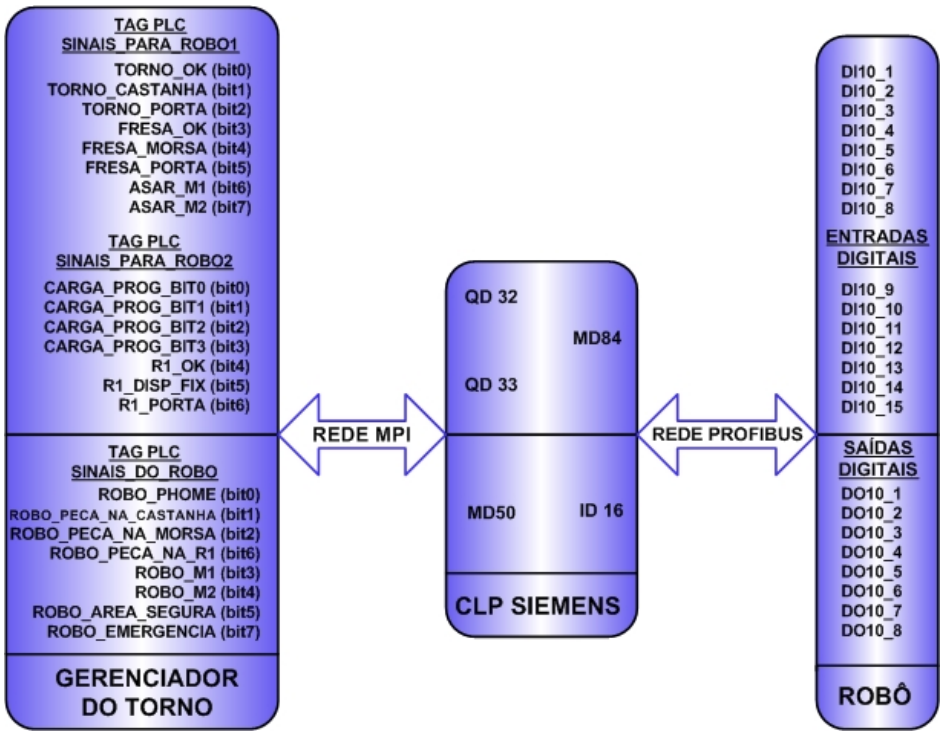


Figura 5.24. Conexões entre gerenciador, CLP Siemens e robô.

5.4.3 Integração entre CLP Siemens e o robô

A conexão entre o CLP Siemens e o controlador do robô se faz por meio de uma rede Profibus DP (Periferia Descentralizada), interligando o CLP com um cartão modelo DSQC 352 instalado no controlador do robô.

O Profibus é baseado em normas internacionalmente reconhecidas. A arquitetura do protocolo é orientada pelo modelo de referência OSI (*Open System Interconnection*) em conformidade com a norma internacional ISO 7498. Neste modelo, todas as camadas geram tarefas de transmissão perfeitamente definidas. A Camada 1 (física) define as características físicas da transmissão. A Camada 2 (ligação de dados) define o protocolo de acesso à rede. A Camada 7 (aplicação) define as funções da aplicação (tabela 5.15).

Tabela 5.15. Arquitetura do protocolo Profibus

Camadas	Tipos de Profibus		
7	FMSp	DP	
3 a 6	NÃO USADA		
2	*Lower Layer Interface* (LLI)	*Direct Data Link Mapper* (DDLM)	IEC Interface
1	RS485 / Fibra Ótica		IEC1158-2

O Perfil de Comunicação Profibus DP, utilizado na conexão CLP-robô, foi projetado para a troca de dados ao nível de campo. O dispositivo central (neste caso o CLP Siemens) comunica-se com os dispositivos de campo distribuídos (neste caso o cartão DSQC 352) através de uma ligação série. Cada dispositivo de campo recebe um endereço na rede. O controlador do robô foi endereçado como dispositivo número 10.

O Profibus DP é um protocolo de comunicação, que usa as camadas 1 e 2 e uma interface de utilizador. As camadas de 3 a 7 não são utilizadas. Este tipo de arquitetura assegura a transmissão rápida e eficiente de dados. O *Direct Data Link Mapper* (DDLM) fornece ao usuário a interface de fácil acesso à camada 2. Tanto as funções de aplicação disponíveis ao usuário, quanto o comportamento do sistema e dos vários tipos de dispositivos DP, são especificados nessa interface. No Profibus FMSp (protocolo universal de comunicações) é dada uma particular importância às camadas 1, 2 e 7. A camada de aplicação 7 consiste na especificação das mensagens da rede (*Fieldbus Message Specification* - FMSp) e da interface de camada mais baixa (L*ower Layer Interface* - LLI). O FMSp

define um grande número de serviços de comunicação mestre-mestre e comunicação mestre-escravo. O LLI define a representação do serviço FMSp no protocolo de transmissão de dados da camada 2. A figura 5.25 mostra a arquitetura do protocolo Profibus (PROFIBUS, 2000).

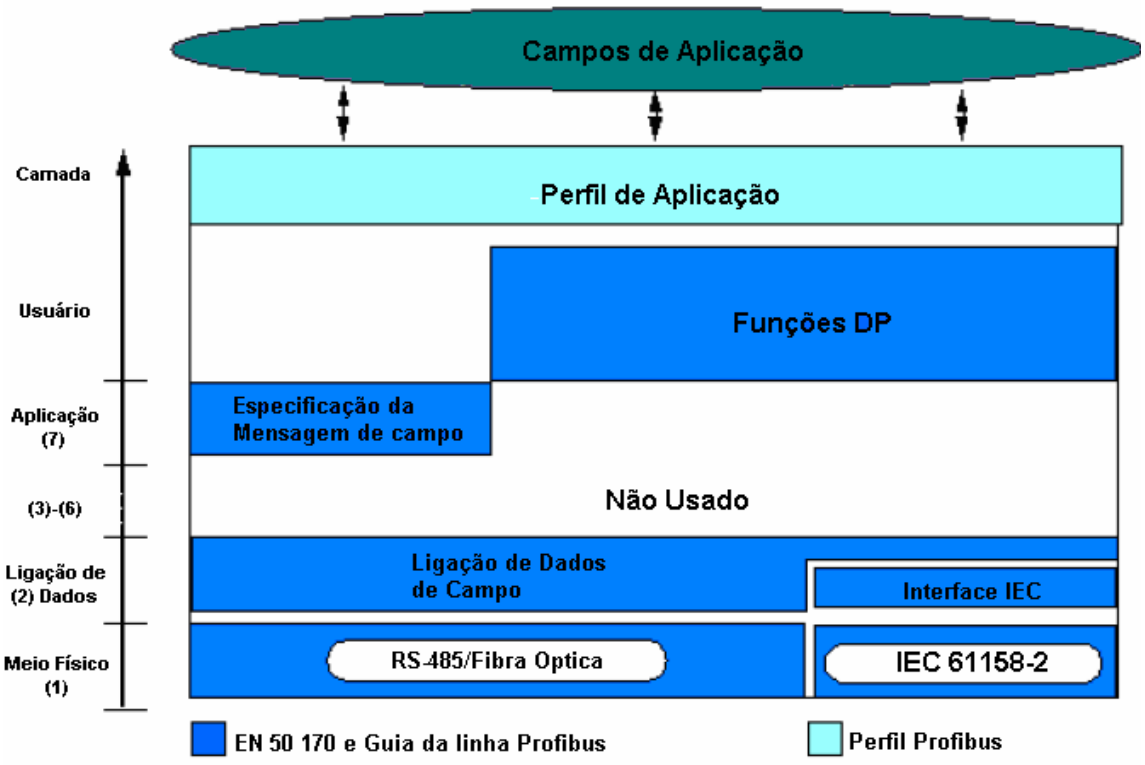


Figura 5.25. Arquitetura do protocolo Profibus.
(Fonte:PROFIBUS, 2000)

A figura 5.24 apresenta as conexões entre o gerenciador do torno, o CLP Siemens e o controlador do robô. As *tags* tipo PLC utilizadas no aplicativo de gerenciador são decompostas em *bits* e têm uma correspondência com entradas e saídas digitais (DI e DO) do controlador do robô. A tabela 5.16 apresenta a relação entre eles.

Tabela 5.16. Relação entre *tags* PLC e I/Os digitais do robô

Gerenciador do torno			I/O digitais do robô	
Tag tipo PLC	Desmembramento da *tag*	*Bit*	Entradas	Saídas
SINAIS_PARA_ROBO1	TORNO_OK	0	DI10_1	
	TORNO_CASTANHA	1	DI10_2	
	TORNO_PORTA	2	DI10_3	
	FRESA_OK	3	DI10_4	
	FRESA_MORSA	4	DI10_5	
	FRESA_PORTA	5	DI10_6	
	ASAR_M1	6	DI10_7	
	ASAR_M2	7	DI10_8	

Gerenciador do torno			I/O digitais do robô	
Tag tipo PLC	Desmembramento da *tag*	*Bit*	Entradas	Saídas
SINAIS_PARA_ROBO2	CARGA_PROG_BIT0	0	DI10_9	
	CARGA_PROG_BIT1	1	DI10_10	
	CARGA_PROG_BIT2	2	DI10_11	
	CARGA_PROG_BIT3	3	DI10_12	
	R1_OK	4	DI10_13	
	R1_DISP_FIX	5	DI10_14	
	R1_PORTA	6	DI10_15	
SINAIS_DO_ROBO	ROBO_PHOME	0		DO10_1
	ROBO_PECA_NA_CASTANHA	1		DO10_2
	ROBO_PECA_NA_MORSA	2		DO10_3
	ROBO_PECA_NA_R1	6		DO10_7
	ROBO_M1	3		DO10_4
	ROBO_M2	4		DO10_5
	ROBO_AREA_SEGURA	5		DO10_6
	ROBO_EMERGENCIA	7		DO10_8

Com base na tabela anterior, que mapeia a relação de entradas e saídas, foi possível construir o programa de movimentação do robô, utilizando a linguagem RAPID, fornecida pela fabricante ABB. A versão utilizada pelo controlador do robô é a S4. O conjunto de quatro *tags* responsáveis pela carga do programa de movimentação do robô (CARGA_PROG_BIT0, ...BIT1, ...BIT2, ...BIT3) está diretamente ligado às entradas digitais DI10_9, _10, _11 e _12, respectivamente. Estas entradas digitais foram configuradas no controlador do robô como um grupo, denominado “Comando”. A partir do grupo Comando construiu-se o programa principal de movimentação do robô (tabela 5.17).

Tabela 5.17. Programa principal do robô

Programa principal do robô			
PROC main()	CASE 4:	f1m2;	CASE 15:
início:	t1m2;	CASE 9:	emer;
TEST Comando	CASE 5:	m1r1;	CASE 0,13,14:
CASE 1:	m1f1;	CASE 10:	GOTO início;
m1t1;	CASE 6:	r1m1;	ENDTEST
CASE 2:	f1m1;	CASE 11:	GOTO início;
t1m1;	CASE 7:	m2r1;	ENDPROC
CASE 3:	m2f1;	CASE 12:	ENDMODULE
m2t1;	CASE 8:	r1m2;	

O programa inicia testando o grupo de entradas denominado “Comando”. Se o grupo, que está ligada à combinação BCD dos quatro primeiros *bits* da *tag* SINAIS_PARA_ROBO2, for igual a um (CASE 1), então o programa executa um desvio para a rotina denominada m1t1, que é o programa do

robô que busca a peça na mesa1 do armazém e a conduz ao torno1. A relação entre os códigos dos programas de movimentação, usinagem, CNC utilizado e mesa de abastecimento e retorno, foi mostrada na tabela 5.13, no item 5.4.1. Se a combinação de entrada for igual a 2 (CASE 2) será executado o programa t1m1, que retira a peça já processada do interior do torno1 e a devolve à mesa1. O programa contempla todas as combinações entre a mesa1 e a mesa2, com os três CNCs disponíveis na célula (torno, fresadora e CNC R1), tanto no carregamento de matéria-prima, quanto no retorno das peças ao armazém. Há doze combinações possíveis, representando doze programas de movimentação. A análise da escolha do programa a ser executado é semelhante para todos eles. A combinação de entrada igual a 15 (CASE 15) está reservada para parada de emergência. A combinação BCD nula na entrada (Comando=0) não executa nenhum desvio para programas de movimentação, mas força o *loop* (retorno) para um novo teste do grupo Comando. Este processo pode se repetir indefinidamente até que haja uma combinação válida na entrada. As combinações BCD 13 e 14 também forçam o retorno ao teste do grupo Comando. Elas não são utilizadas pelo sistema e podem ser consideradas como combinações de reserva para futuras ampliações.

Dos doze programas de movimentação do robô, serão analisados em detalhes somente os de carregamento do torno, a partir da mesa1 e o descarregamento correspondente. A totalidade dos programas de movimentação do robô se encontra no Apêndice C desta dissertação. A tabela 5.18 mostra os programas m1t1 (da mesa1 para o torno1) e t1m1 (do torno1 para a mesa1).

Tabela 5.18. Programas de carga e descarga do torno a partir da mesa1

	Programa mesa1-torno1	Programa torno1-mesa1
Nº	PROC m1t1()	PROC t1m1()
1	!Verifica se o torno está OK	!Verifica se o torno está OK
2	IF DI10_1=1 THEN	IF DI10_1=1 THEN
3	! início do programa mesa1-torno1	! início do programa torno1-mesa1
4	MoveAbsJ jposhome,v200,z80,tool0;	! espera sinal de porta aberta
5	MoveJ p1,v200,z80,tool0;	WaitDI DI10_3,1;
6	! informa que robô está fora de Phome	SingArea\Wrist;
7	Reset DO10_1;	MoveJ p16,v200,z100,tool0;
8	MoveJ p2,v200,z80,tool0;	! informa que robô está fora de Phome
9	! informa que robô está fora de Área Segura	Reset DO10_1;
10	Reset DO10_6;	MoveL p5,v200,fine,tool0;
11	MoveL p3,v100,fine,tool0;	! informa que robô está fora de Área Segura
12	WaitTime 1;	Reset DO10_6;
13	! fecha a garra para pegar a peça no palete da	MoveL p6,v200,fine,tool0;
14	Set DO11_8;	MoveL p4,v80,fine,tool0;
15	WaitTime 1;	WaitTime 1;
16	MoveJ p2,v200,z80,tool0;	! fecha a garra para pegar peça

	Programa mesa1-torno1	Programa torno1-mesa1
17	SingArea\Wrist;	Set DO11_8;
18	MoveJ p16,v200,fine,tool0;	WaitTime 1;
19	MoveL p5,v200,fine,tool0;	! envia pulso para abrir castanha
20	! espera sinal de confirmação da porta do	Set DO10_2;
21	WaitDI DI10_3,1;	! espera sinal de castanha aberta
22	! espera sinal de confirmação da castanha do	WaitDI DI10_2,0;
23	WaitDI DI10_2,0;	! encerra pulso para fechar a castanha
24	MoveL p6,v200,fine,tool0;	Reset DO10_2;
25	MoveL p4,v80,fine,tool0;	MoveL p6,v200,fine,tool0;
26	! envia sinal para fechar a castanha (pulso)	MoveL p5,v200,fine,tool0;
27	Set DO10_2;	! envia sinal de robô em área segura
28	! espera sinal de castanha fechada	Set DO10_6;
29	WaitDI DI10_2,1;	MoveJ p16,v200,z100,tool0;
30	! encerra pulso para fechar a castanha	MoveJ p2,v200,z100,tool0;
31	Reset DO10_2;	MoveL p3,v100,fine,tool0;
32	! abre a garra para deixar a peça no torno	WaitTime 1;
33	Reset DO11_8;	! devolve a peça para o palete 1
34	WaitTime 1;	Reset DO11_8;
35	MoveL p6,v200,fine,tool0;	WaitTime 1;
36	MoveL p5,v200,fine,tool0;	MoveJ p2,v200,z80,tool0;
37	! envia sinal de robô em área segura	! envia sinal ao AS/AR para retorno do
38	Set DO10_6;	Set DO10_4;
39	MoveAbsJ jposhome,v200,z80,tool0;	MoveJ p1,v200,z80,tool0;
40	! envia sinal de robô em Phome	MoveAbsJ jposhome,v200,z80,tool0;
41	Set DO10_1;	! envia sinal de robô em Phome
42	ENDIF	Set DO10_1;
43	ENDPROC	! aguarda confirmação do AS/AR de retorno
44		WaitDI DI10_7,0;
45		! encerra pulso para retorno da M1
46		Reset DO10_4;
47		ENDIF
48		ENDPROC

A coluna à esquerda contém uma referência numérica para facilitar a localização de determinadas linhas. Elas não fazem parte da lista de programação. Os programas contêm comentários (precedidos do símbolo !) descrevendo passo-a-passo todos os eventos relevantes na movimentação do robô. Pode-se afirmar que a seqüência de eventos dos programas é uma representação fiel da modelagem em RdPI.

O programa m1t1 é o responsável por buscar a matéria-prima que chegou na mesa1 dos berços de usinagem do armazém e conduzi-la ao torno para processamento. A primeira instrução executada após a carga do programa é verificar se o torno carregou o programa de usinagem e abriu a porta (IF DI10_1=1). Em caso afirmativo, o robô inicia sua movimentação, iniciando pelo ponto P1 de onde

informa que já não se encontra mais em Phome (linha 7 - Reset DO10_1). Esta informação sinaliza ao sistema que o robô não está mais disponível, mas executando um programa de movimentação. Na seqüência o robô se desloca até o ponto P2 e informa que já não está mais em área segura (linha 10 - Reset DO10_6), pois se aproxima do *pallet* da mesa1. Ao chegar ao ponto P3 (no *pallet* da mesa1), o robô fecha a garra sobre a peça (linha 14 - Set DO11_8) e a conduz em direção ao torno através dos pontos P2, P16 e P5 (na frente da porta do torno), onde aguarda a confirmação redundante de que a porta do torno está aberta (linha 21 - WaitDI DI10_3,1) e que a castanha também está aberta (linha 23 – WaitDI10_2,0). Atendidas as duas condições, o robô entra no torno seguindo os pontos P6 e P4 (posição da peça dentro da castanha). Nesse momento o robô indica que a peça está em posição na castanha (linha 27 - Set DO10_2). Este comando autoriza o torno a fechar a castanha. O robô aguarda a confirmação de que a castanha está efetivamente fechada (linha 29 - WaitDI DI10_2,1) para encerrar a ordem anterior (linha 31 - Reset DO10_2). Após o fechamento da castanha, o robô abre a garra (linha 33 - Reset DO11_8) deixando a peça no torno e sai do mesmo, seguindo os pontos P6, P5 (na frente da porta do torno), onde informa que se encontra novamente em área segura (linha 38 - Set DO10_6). Este sinal será a indicação para o torno fechar a porta e iniciar o ciclo de usinagem. Finalmente o robô retorna ao seu ponto de origem, Phome, e comunica que está novamente disponível para executar outro programa (linha 41 - Set DO10_1). Ao fim do programa m1t1, há um retorno automático ao programa principal, que lê ciclicamente o grupo Comando a espera de uma instrução válida, para executar o próximo programa.

O programa t1m1 executa o retorno do material já processado no torno para a mesa1. A primeira instrução do programa verifica se o torno finalizou o ciclo de usinagem (IF DI10_1=1). Se esta condição foi atendida, então o programa aguarda a indicação de que a porta do torno está realmente aberta (linha 5 - WaitDI DI10_3,1). Ao receber a confirmação, o robô inicia seu deslocamento em direção ao torno indo até o ponto P16, onde informa que saiu de Phome (linha 9 – Reset DO10_1) e ponto P5, onde sinaliza que já não se encontra mais em área segura (linha 12 - Reset DO10_6). O robô continua seu deslocamento através dos pontos P6 e P4 (posição na castanha), onde fecha a garra sobre a peça já processada (linha 17 – Set DO11_8). Nesse momento o robô necessita enviar um pulso para abrir a castanha do torno (linha 20 – Set DO10_2). Quando a confirmação da abertura da castanha é recebida (linha 22 – Wait DI10_2,0), o robô encerra o pulso de abertura (linha 24 – Reset DO10_2). Com a peça presa na garra e a castanha do torno aberta, o robô inicia a retirada através dos pontos P6 e P5 (ponto na frente da porta do torno) onde informa que retornou a uma área segura (linha 28 – Set DO10_6). O caminho do robô em direção ao *pallet* da mesa1 continua pelos

pontos P16, P2 e P3 (no *pallet*). Ao chegar ao *pallet*, o robô abre a garra para soltar a peça (linha34 – Reset DO11_8) e inicia seu retorno para Phome, iniciando pelo ponto P2, de onde envia um sinal para que o armazém retorne o *pallet* da mesa1 (linha 38 – Set DO10_4). Na seqüência da movimentação estão os pontos P1 e, finalmente Phome, onde o robô envia a informação de que está novamente disponível para executar outro programa (linha 42 – Set DO10_1). Antes de encerrar o programa, o robô necessita receber a confirmação do retorno do *pallet* da mesa1 ao armazém (linha 44 – WaitDI DI10_7,0) para encerrar o pulso que comanda o retorno (linha 46 – Reset DO10_4).

A figura 5.26 mostra as posições dos pontos da trajetória do robô, utilizados na programação, assim como a interface de usuário do gerenciador, o CLP Siemens, o controlador do robô e o *Teach Pendant.* Uma tabela contendo a sequência de pontos de todos os programas de movimentação do robô se encontra no Apêndice C desta dissertação.

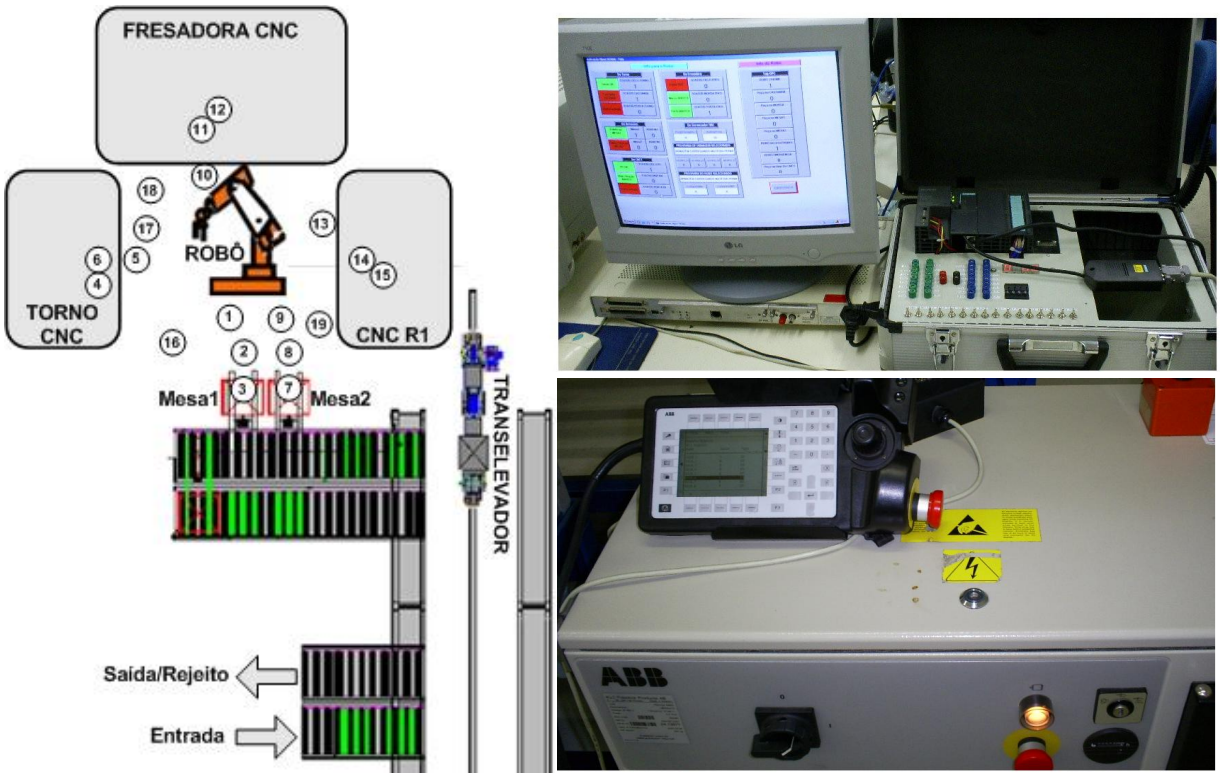


Figura 5.26. Posições da trajetória do robô, CLP Siemens e controlador do robô.

5.4.4 Integração do gerenciador, CLP Siemens e robô

A integração do aplicativo gerenciador do torno, o CLP Siemens e o robô, permite que o robô execute suas funções de carregamento e descarregamento dos CNC com grande autonomia. Há uma

preocupação constante pela segurança operacional do robô, pois ele é o recurso da célula com maior potencial de risco, principalmente de colisão. Os programas de movimentação contém redundâncias no intertravamento com as portas e dispositivos de fixação dos CNCs. A arquitetura do programa desenvolvido para o robô permite que ele não fique preso à operação de usinagem. Após a carga da matéria-prima nos CNCs, o robô retorna ao Phome, ficando disponível para a futura operação de descarga correspondente, ou qualquer operação solicitada por outro CNC da célula. A seqüência de atendimento às solicitações é determinada exclusivamente pela ordem temporal de chegada dos pedidos.

O disparo dos programas de carga dos equipamentos CNC é realizado pela atualização da *tag* AuxiliarProg, ligada por meio de *script* à *tag* ProgUsinagem, conforme mostrado pela tabela 5.13. Os programas de retorno das peças processadas nos CNCs é disparada pela abertura da porta dos CNCs, ação que indica o fim do ciclo de usinagem. A figura 5.27 mostra o *script* associado à *tag* OPC ComunicTorno_Robô que aciona a descarga de peça no torno. Após a figura, a tabela 5.19 estabelece a relação entre a *tag* AuxiliarProg, a mesa para onde a peça deverá retornar após o processo de usinagem e o programa do robô que executa a movimentação correspondente.

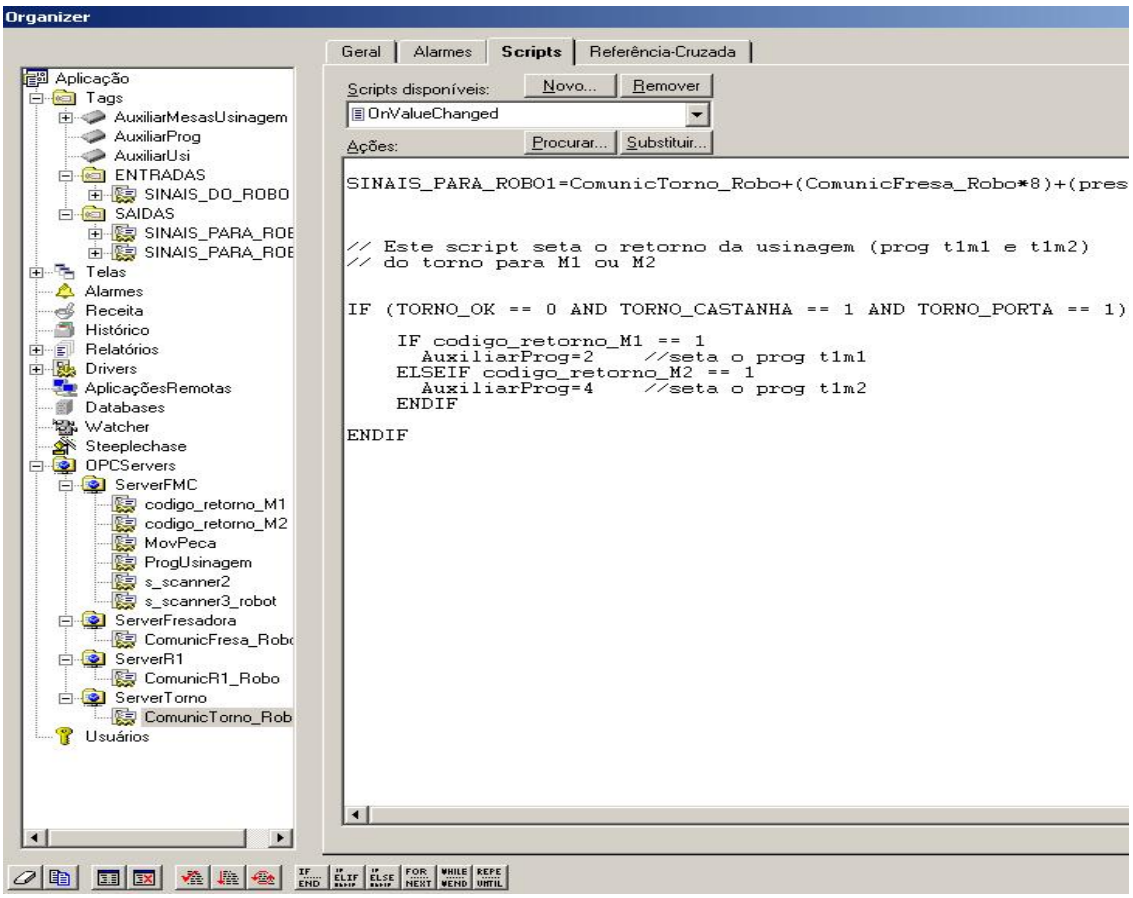


Figura 5.27. *Script* retorno da peça da usinagem

Tabela 5.19. Relação entre a *tag* AuxiliarProg, mesa de usinagem e programa do robô

AuxiliarProg	Mesa	Programa do Robô
0	-	-
1	1	M1T1
7	2	M2F1
5	1	M1F1
3	2	M2T1
1	1	M1T1
7	2	M2F1
9	1	M1R1
11	2	M2R1
2	1	T1M1
4	2	T1M2
6	1	F1M1
8	2	F1M2
10	1	R1M1
12	2	R1M2

Após o início da movimentação do robô para retornar a peça ao *pallet* da mesa de usinagem, é necessário zerar a *tag* AuxiliarProg, pois se ela estiver com o mesmo conteúdo quando o robô finalizar o movimento, o robô voltará a executar o mesmo programa indevidamente. O momento em que o robô pega a peça no dispositivo de fixação do CNC, é o evento que atualiza a *tag* (AuxiliarProg =0), através de um *script* associado à *tag* ROBO_PECA_NA_CASTANHA. Todos os *scripts* utilizados no desenvolvimento da célula podem ser consultados no Apêndice D desta dissertação.

5.5 GERENCIADOR DO ARMAZÉM E GERENCIADOR FMC

O armazém automático já possuía uma interface com o usuário, desenvolvida no *software* Elipse SCADA, antes do início do projeto. Este aplicativo foi alterado e ampliado para cumprir o papel de gerenciador do armazém e da FMC. Esta decisão foi tomada no intuito de simplificar a construção do gerenciador FMC, já que o aplicativo que comanda o armazém automático necessita de poucas adaptações para tornar-se o gerente desejado. Toda a estrutura de programação de *scripts*, CLP e Banco de Dados (BD) será aproveitada. Seria inviável construir integralmente um novo aplicativo para o armazém dentro da proposta deste trabalho, dada a complexidade do mesmo. As modificações introduzidas no aplicativo Elipse original permitem que o armazém troque informações com o gerenciador do robô e o gerenciador FMC. Este capítulo abordará somente as modificações que foram

introduzidas no *software* do armazém. Informações adicionais a respeito do funcionamento geral do armazém automático (AS/RS - *Automatic Storage/Retrieval System*) fabricado pela Scheffer, poderão ser obtidas em duas monografias do IST, que descrevem em detalhes *hardware* e *software* do equipamento (KLUG, 2001 e SANTOS, 2003).

O AS/RS tem a função de armazenar matéria-prima, acondicionada em *pallets*; retirar os *pallets* solicitados das prateleiras; encaminhar os *pallets* para as mesas nos berços de usinagem; e retornar os *pallets* das mesas nos berços de usinagem. A figura 5.28 apresenta vistas parciais do armazém.

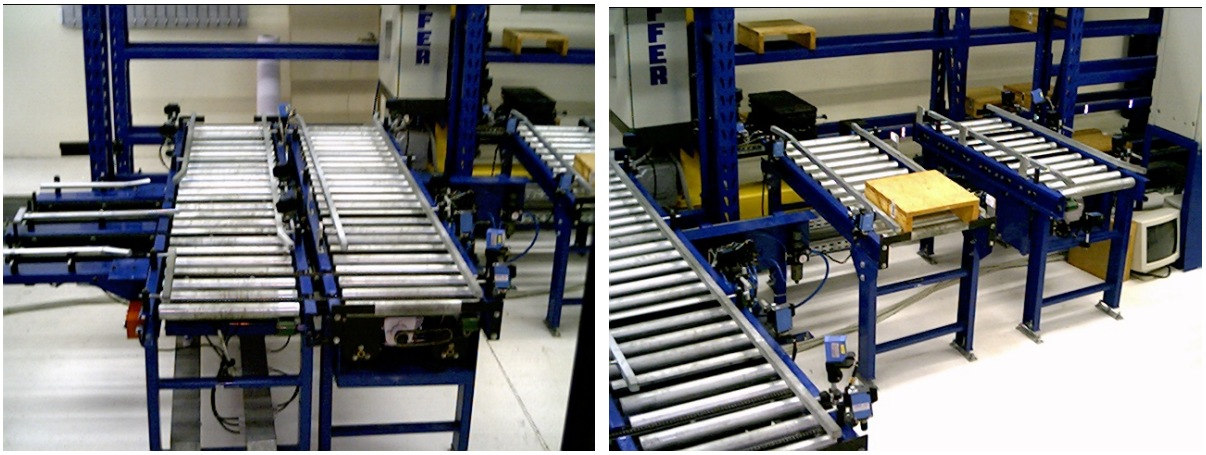


Figura 5.28. Berços de usinagem, saída/rejeito e entrada.

A figura da esquerda mostra os berços de usinagem, com a mesa1 e mesa2. A figura da direita mostra parcialmente o berço de usinagem; ao centro, o berço de saída ou rejeito (com um *pallet*); à direita, o berço de entrada de *pallets*. Ao fundo tem-se uma visão parcial da estrutura porta-*pallet* e do transelevador

Recuperando as interfaces definidas para os gerenciadores do armazém e da FMS, temos:

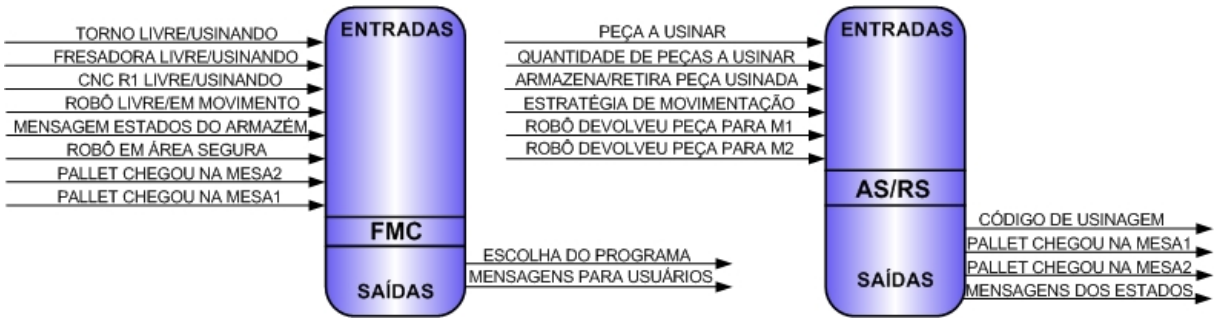


Figura 5.29. Interfaces para os gerenciadores FMC e do armazém.

Dentro da proposta de construir os dois gerenciadores dentro de um único aplicativo, podemos unificar as interfaces de entrada e saída. A conversão para *tags*, que é o primeiro passo para a construção do aplicativo, é mostrada pela tabela 5.20.

Tabela 5.20. Conversão de interfaces do armazém e FMC para *tags*

INTERFACES	*TAG*
ROBÔ LIVRE/EM MOVIMENTO	ROBO_PHOME
ROBÔ EM ÁREA SEGURA	ROBO_AREA_SEGURA
FRESADORA LIVRE/USINANDO	FRESA_OK
TORNO LIVRE/USINANDO	TORNO_OK
CNC R1 LIVRE/USINANDO	R1_OK
MENSAGENS DOS ESTADOS DO ARMAZÉM	Mensagem
PALLET CHEGOU NA MESA1	presença_mesa1
PALLET CHEGOU NA MESA2	presença_mesa2
ESCOLHA DO PROGRAMA	ProgUsinagem
PEÇA A USINAR	MovPeca
QUANTIDADE DE PEÇAS A USINAR	MovUnidades
ARMAZENA/RETIRA PEÇA USINADA	MovRetSaida
ESTRATÉGIA DE MOVIMENTAÇÃO	Estrategia1, 2 e 3
ROBÔ DEVOLVEU PEÇA PARA M1	ROBÔ_M1
ROBÔ DEVOLVEU PEÇA PARA M2	ROBÔ_M2

5.5.1 Aplicativo gerenciador do armazém e da FMC no Elipse SCADA

Com base na definição das *tags* necessárias e contando com o aplicativo original do armazém, iniciou-se a construção dos gerenciadores da FMC e do AS/RS. Apesar de fazerem parte do mesmo aplicativo, a função desempenhada por cada gerenciador é bem definida. O gerenciador do AS/RS deve comandar as ações de entrada de *pallet* contendo matéria-prima ou peças, armazenamento dos *pallets*, condução de peças para os berços de usinagem, retorno das peças dos berços de usinagem e retirada de *pallets*. A necessidade da intervenção de um operador se resume ao abastecimento de material nos *pallets* e a colocação dos mesmos na esteira de entrada. Esta tarefa também pode ser realizada por um AGV (Veículo Auto-Guiado), inclusive o aplicativo do armazém prevê esta condição, porém ele não será utilizado porque o laboratório não conta com este recurso. Ao realizar a entrada de uma matéria-prima no armazém, o operador necessita informar um código, referente ao tipo de material a ser armazenado. Estes códigos são previamente cadastrados no banco de dados do gerenciador, assim como o código de usinagem, que precisa de uma matéria-prima correspondente e

uma mesa de usinagem pré-determinada. O banco de dados utilizado é o Access 2003, da Microsoft. A figura 5.30 apresenta em detalhes o cadastro de peças e processos no gerenciador do armazém.

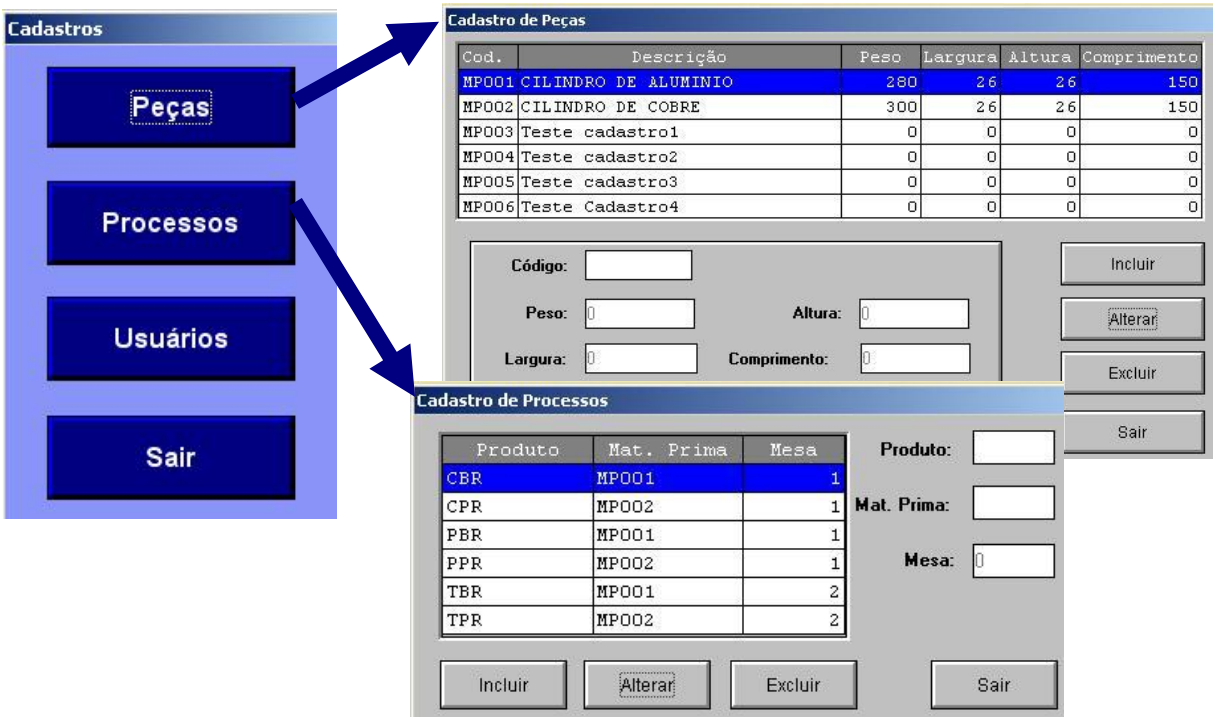


Figura 5.30. Cadastro de peças e processos no gerenciador do armazém.

A peça descrita como “Cilindro de Alumínio” foi cadastrada no BD do armazém com o código “MP001”, na tabela de “Cadastro de Peças”. Ela será a matéria-prima para fabricação do produto “CBR” (cavalo branco), que utilizará a mesa1 dos berços de usinagem, conforme a tabela “Cadastro de Processos”. Todas as peças brancas do jogo de xadrez serão produzidas a partir da matéria-prima “MP001” Já as peças pretas serão feitas da matéria-prima “MP002”. Sempre que houver um acréscimo na família de peças que a célula irá fabricar, será necessário atualizar os cadastros de peças e processos.

Para solicitar a fabricação de uma peça o usuário da célula deve inserir o código da peça pronta (CBR, por exemplo), indicar quantas unidades da peça serão produzidas, para onde será conduzida a peça após o retorno da usinagem e a prioridade da operação (normal ou urgente). A figura 5.31 mostra a tela de movimentação e um pedido de fabricação na célula. A tabela 5.21 fornece a relação de peças cadastradas para a célula, suas respectivas matérias-primas e mesas de usinagem, além do CNC que efetuará o processamento.

Figura 5.31. Tela de solicitação de usinagem.

Tabela 5.21. Relação de código de peças

ProgUsinagem	Matéria-prima	Mesa	CNC
PBR	MP001	1	Torno
PPR	MP002	1	Torno
TBR	MP001	2	Fresadora
TPR	MP002	2	Fresadora
CBR	MP001	1	Fresadora
CPR	MP002	1	Fresadora
BBR	MP001	2	Torno
BPR	MP002	2	Torno
DBR	MP001	1	Torno
DPR	MP002	1	Torno
RBR	MP001	2	Fresadora
RPR	MP002	2	Fresadora
M1R1	MP001	1	CNC R1
M2R1	MP002	2	CNC R2

A *tag* MovPeca, do gerenciador do armazém, está associada a um objeto tipo *Setpoint* onde o usuário digita o código da peça a ser usinada. No momento em que o usuário aperta a tecla "Confirmar", o conteúdo da *tag* MovPeca é copiado para várias tabelas do Banco de Dados. Quando a tarefa for executada, e o *pallet* chegar a uma das mesas dos berços de usinagem, a *tag* ProgUsinagem é carregada com o valor da *tag* MovPeca, através de uma leitura do BD. A caixa "Saída" seleciona, indica que o *pallet* será conduzido ao berço de saída, e não às prateleiras de armazenagem, ao fim do processo de usinagem. Utilizando como exemplo a solicitação apresentada na figura 5.31, temos que, para produzir uma peça do cavalo branco (CBR), utiliza-se a matéria-prima MP001. O *pallet* será levado pelo armazém até a mesa1 e o robô conduzirá a peça até a fresadora. A figura 5.32 mostra o *script* associado à *tag* presença_mesa1 (acionada pelo evento da chegada do *pallet* na mesa1), associada ao evento OnValueChanged. Este *script* atualiza a *tag* ProgUsinagem que será lida por todos os equipamentos da célula via padrão de comunicação OPC. Decorrente da leitura desta *tag*, o gerenciador do robô carregará o programa de movimentação m1f1 (da mesa1 para a fresadora1) e a fresadora também carregará o programa correspondente de usinagem.

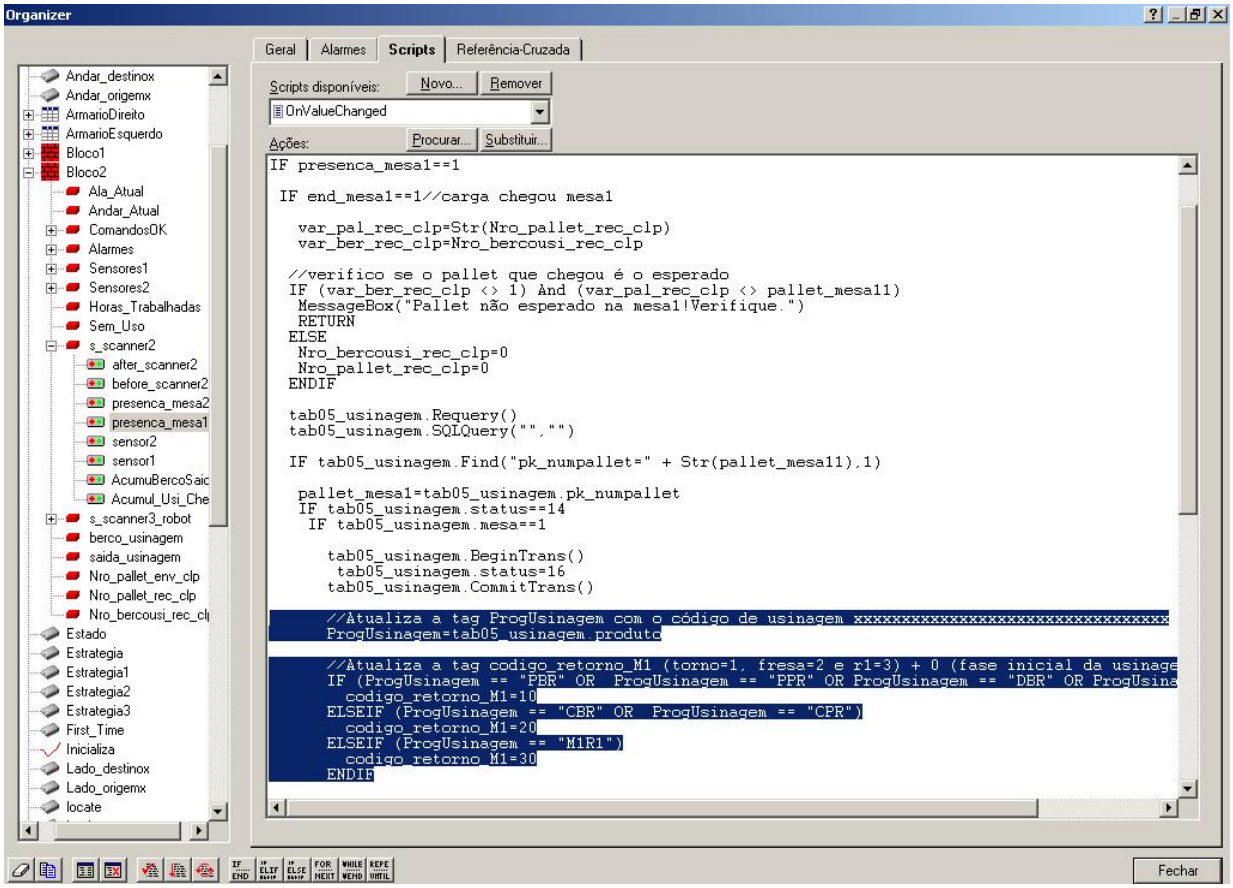


Figura 5.32. *Script* associado à *tag* presença_mesa1

A instrução do *script* "ProgUsinagem=tab05_usinagem.produto" é a responsável por atualizar a *tag* ProgUsinagem a partir da tab05_usinagem, coluna "produto" do BD. A tabela usinagem do BD é utilizada pelo gerenciador do armazém para seqüenciar a lista de tarefas de usinagem. O usuário do sistema pode solicitar vários pedidos de usinagem ao mesmo tempo. Cada solicitação será inserida na lista de tarefas da tabela. O artifício de atualizar a *tag* ProgUsinagem, a partir da tabela de usinagem, permite sincronizar a movimentação do robô e a carga do programa nos CNCs com a chegada do *pallet* contendo a matéria-prima nas mesas de usinagem. A *tag* código_retorno_M1 recebe um valor numérico que indica de qual equipamento CNC a peça retornará para a mesa1 e em que fase do processo a peça se encontra. O primeiro dígito indica o CNC que está processando o pedido (torno=1, fresadora=2 e CNC R1=3). Se o segundo dígito for igual a zero, significa que a peça está sendo carregada no CNC. No exemplo adotado, se a *tag* ProgUsinagem for igual a CBR, então a *tag* código_retorno_M1=20 (peça está sendo carregada na fresadora). Se o segundo dígito estiver ausente, significa que o processo de usinagem foi iniciado ou que o robô já está em movimento para devolver a peça ao *pallet* da mesa1. Após a atualização da *tag* ProgUsinagem, ocasionada pela chegada do *pallet*

na mesa1, o robô executa o programa m1f1 e conduz a matéria-prima até a fresadora para fabricação do cavalo branco. Quando a usinagem da peça for iniciada, a *tag* ProgUsinagem deve imediatamente ser atualizada, sob risco do programa de movimentação do robô repetir a solicitação inicial. Para cumprir esta função, utiliza-se o fechamento da porta da fresadora (que inicia o ciclo de usinagem) para atualizar a *tag* ProgUsinagem. Desta forma, quando o robô retornar para Phome, ele estará apto a executar uma nova solicitação de movimentação. A figura 5.33 apresenta as *tags* envolvidas e o *script* responsável pela indicação de que a fresadora foi carregada com sucesso e iniciou o ciclo de usinagem.

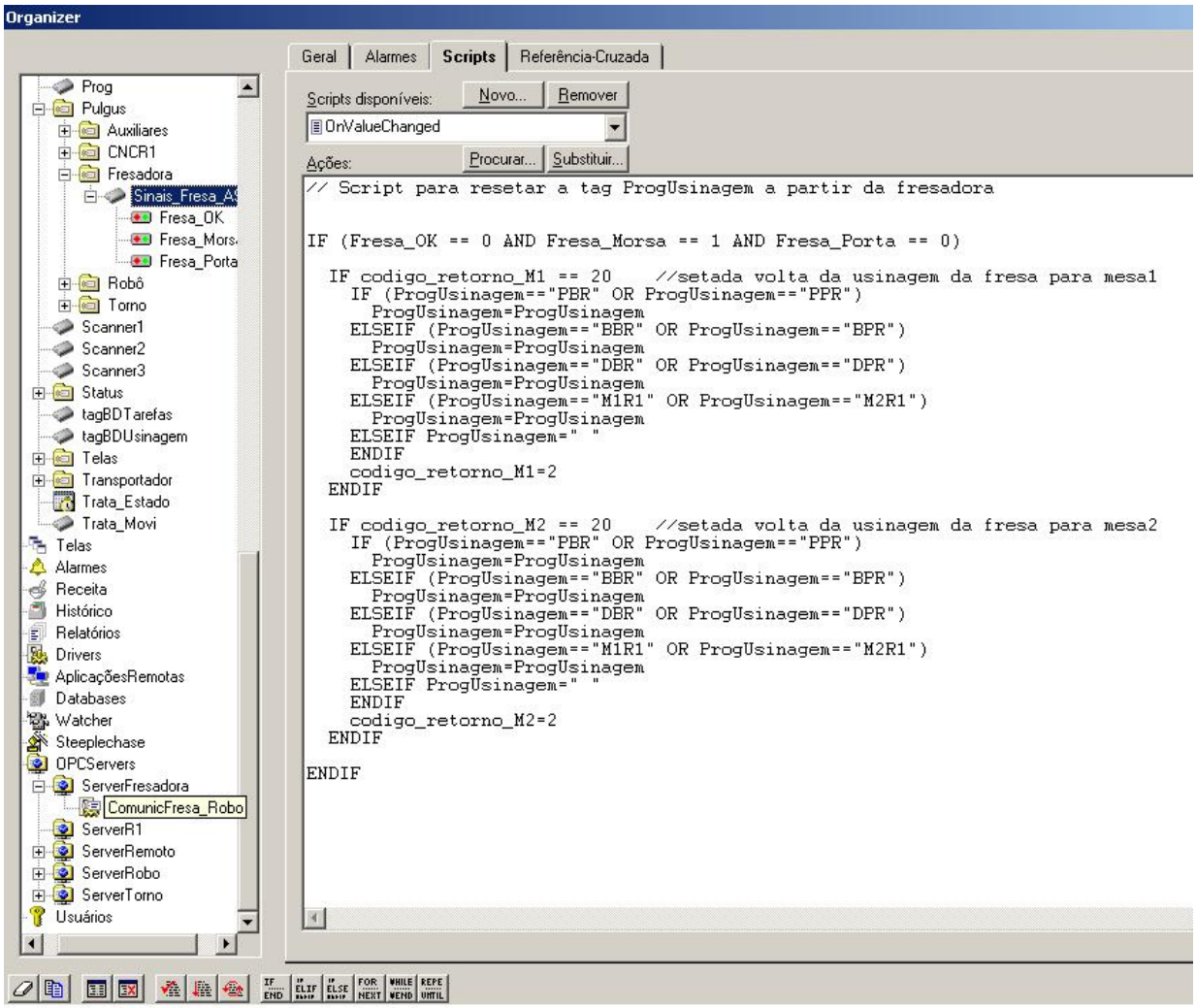


Figura 5.33. *Script* para atualização da *tag* ProgUsinagem após o início da usinagem.

O *script*, que é disparado pelo evento OnValueChanged, monitora cada mudança de valor na *tag* que agrupa os estados da fresadora. Sempre que houver uma mudança na tag, o *script* verifica se a porta da fresadora está fechada (Fresa_Porta=0), se o torno está em ciclo (Fresa_OK=0) e se a

castanha está fechada (Fresa_Morsa=1). Bastaria monitorar a condição da porta somente, mas optou-se pela redundância para aumentar o nível de segurança do sistema. Satisfeita a condição inicial, o *script* verifica se o código de retorno da peça é da fresadora para a mesa1, ou para a mesa2, para então limpar a *tag* ProgUsinagem (ProgUsinagem=" ") e retirar o segundo dígito (zero) da *tag* codigo_retorno_M1 ou codigo_retorno_M2 (de 20 para 2). Esta mudança, apontada pelo código de retorno, indica que a peça iniciou o processo de usinagem. Se, paralelamente houver outra solicitação de usinagem em curso, para o torno ou para o CNC R1, o *script* preserva o conteúdo da *tag* ProgUsinagem.

O procedimento para iniciar o movimento de descarga da peça, ao fim do processo de usinagem, é uma atribuição do gerenciador do robô e foi descrito no item 5.4.4. Quando o robô depositar a peça já usinada de volta no *pallet* correspondente (por exemplo, o da mesa1), a *tag* codigo_retorno_M1 deve ser zerada, indicando o fim do processo de descarga e o início do retorno do *pallet* ao armazém. Este procedimento é executado por um *script* OnValueChanged associado à *tag* interna do AS/RS retorno_robot_mesa1 e pode ser consultada no Apêndice D.

O gerenciador do armazém, assim como o gerenciador FMC, necessitam de diversas informações dos demais gerenciadores, conforme definido pelas interfaces. A figura 5.34 mostra a estrutura dos servidores OPC no *organizer* do aplicativo e uma das telas do gerenciador FMC.

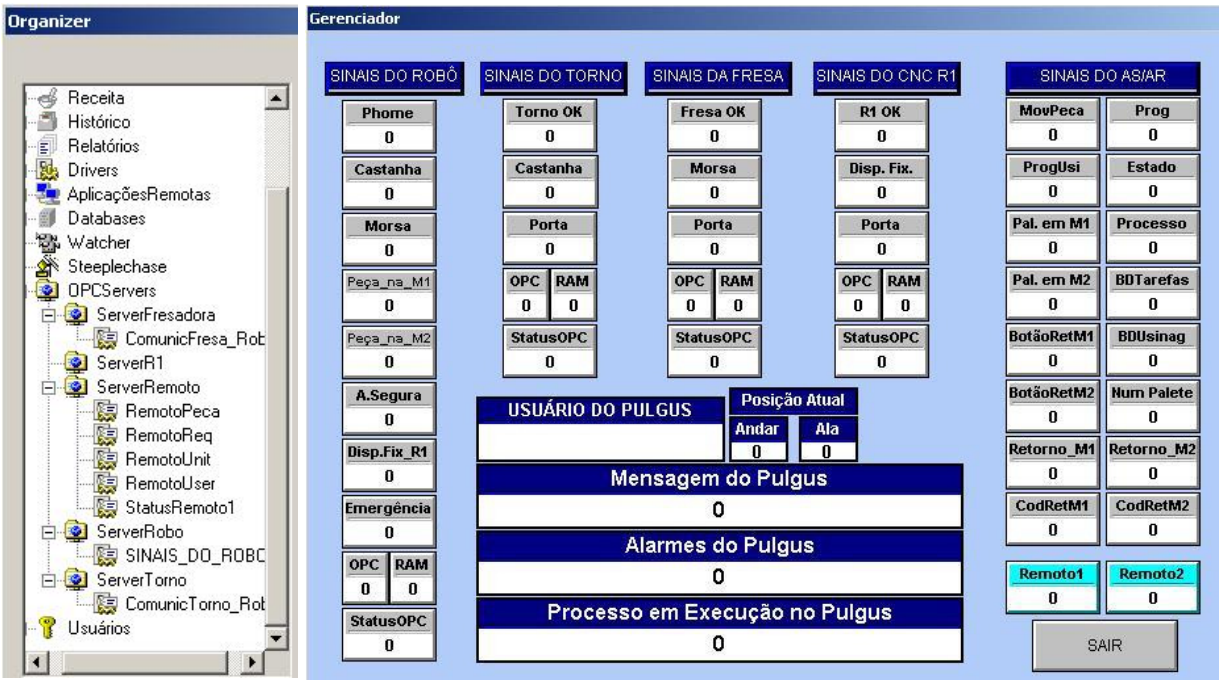


Figura 5.34. Estruturas dos servidores OPC e uma tela do gerenciador FMC

A tela “Gerenciador” tem a função de oferecer ao usuário do sistema um resumo dos estados dos recursos da célula. As informações dos equipamentos CNC, do robô e do gerenciador remoto são coletadas dos servidores OPC. Já as do armazém estão diretamente ligadas às *tags* do gerenciador do AS/RS. As *tags* OPC são decompostas através de *scripts* e organizadas em um grupo de *tags* (Pulgus), que contém os seguintes grupos de *tag* secundários: Auxiliares, CNCR1, Fresadora, Robô e Torno. A figura 5.35 mostra os detalhes da estrutura das *tags* decompostas dos diversos servidores OPC e utilizadas pelos gerenciadores FMC e do armazém. A figura também apresenta a tela principal do aplicativo que contém os gerenciadores.

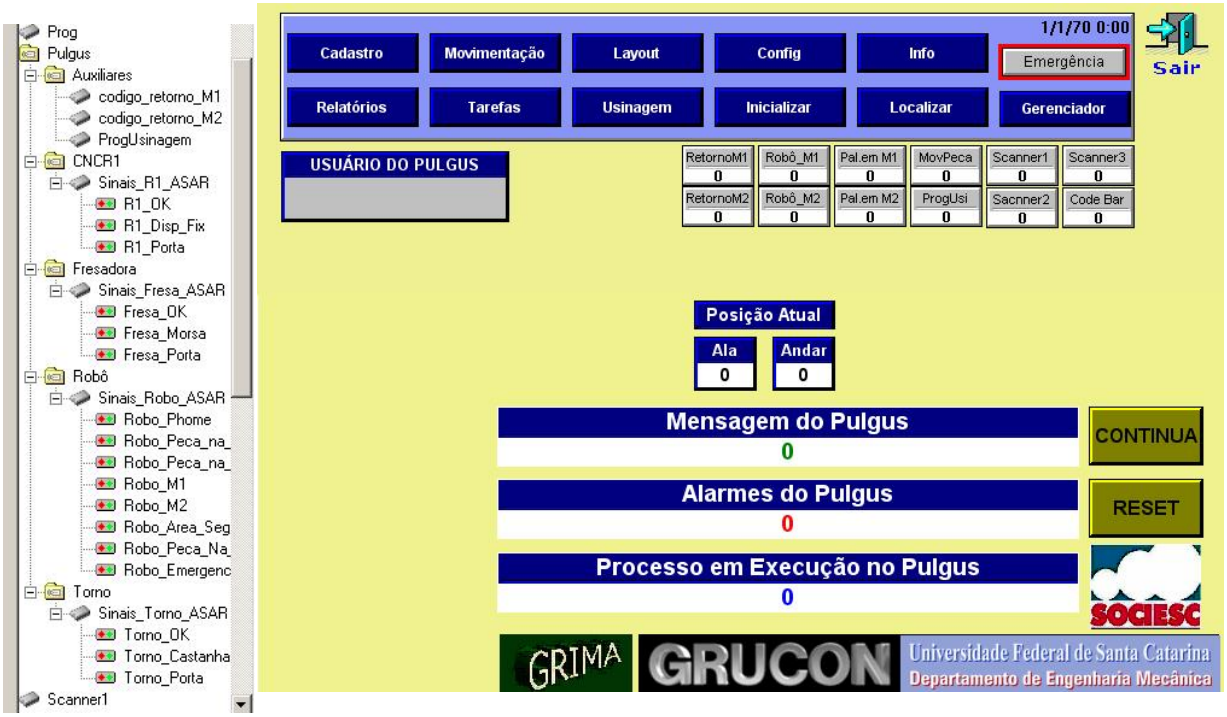


Figura 5.35. Estrutura desmembrada das *tags* OPC e tela principal do aplicativo.

5.6 GERENCIADOR REMOTO

A função do gerenciador remoto é permitir que um usuário remoto possa inserir seu pedido de fabricação de peças na célula (no gerenciador FMC). É necessário que este usuário se encontre dentro da rede local de comunicação. A conexão entre o gerenciador remoto e a FMC é feita via padrão de comunicação OPC.

A definição das mínimas interfaces necessárias para o gerenciador remoto é apresentada pela figura 5.36.

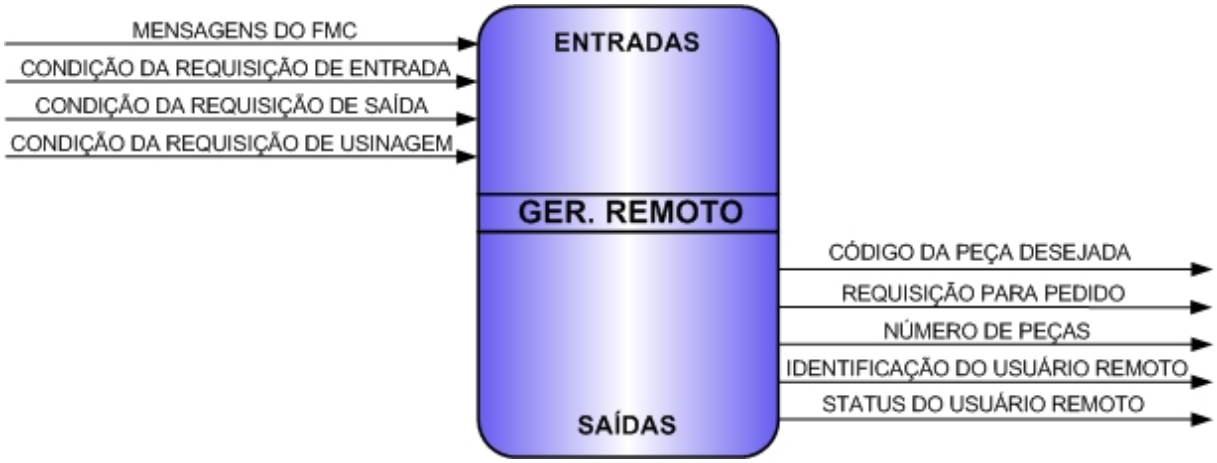


Figura 5.36. Interfaces do gerenciador remoto.

A transformação das interfaces em *tags*, que serão utilizadas para construção do aplicativo no *software* Elipse SCADA, estão na tabela 5.22.

Tabela 5.22. Conversão de interfaces do gerenciador remoto para *tags*

INTERFACES	*TAG*
MENSAGENS DO FMC	Mensagem
CONDIÇÃO DA REQUISIÇÃO DE ENTRADA NO FMC	MovEntrada
CONDIÇÃO DA REQUISIÇÃO DE SAÍDA NO FMC	MovSaída
CONDIÇÃO DA REQUISIÇÃO DE USINAGEM NO FMC	MovUsinagem
CÓDIGO DA PEÇA DESEJADA	RemotoPeca
NÚMERO DE PEÇAS	RemotoUnit
IDENTIFICAÇÃO DO USUÁRIO REMOTO	RemotoUser
STATUS DO USUÁRIO REMOTO	StatusRemoto1
REQUISIÇÃO PARA PEDIDO	RemotoReq

5.6.1 Gerenciador remoto no Elipse SCADA

A construção do gerenciador remoto utiliza as *tags* pré-definidas a partir das interfaces. A figura 5.37 apresenta a estrutura das *tags* no *organizer* e a interface com o usuário.

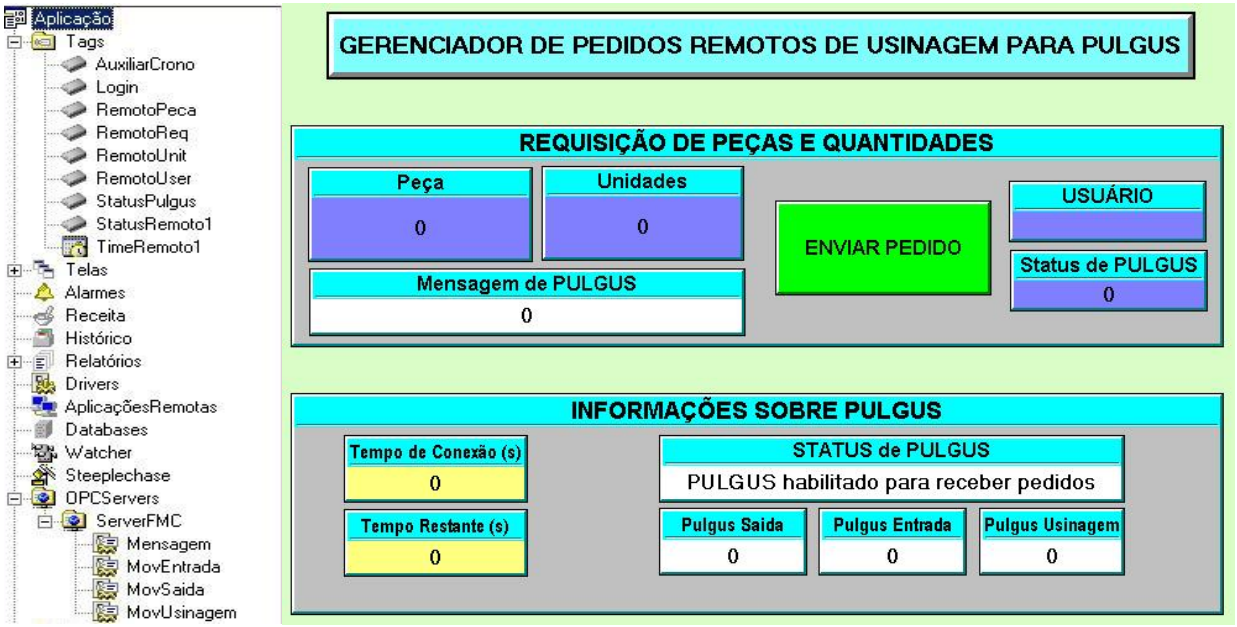


Figura 5.37. Estrutura das *tags* do gerenciador remoto e interface com usuário.

Na tela da interface o usuário remoto digita o código e o número de peças desejadas. Dois objetos tipo *Setpoint* associados às *tags* RemotoPeca e RemotoUnit registram a solicitação. O código da peça solicitada deve constar no cadastro de peças do gerenciador FMC. Caso a peça não esteja cadastrada, o usuário receberá uma mensagem negando o pedido de fabricação. Estas *tags* estão diretamente ligadas às *tags* MovPeca e MovUnidades na tela de movimentação do gerenciador FMC, através do padrão de comunicação OPC e o uso de *script* de programação. Após inserir o pedido o usuário pressiona o botão "ENVIAR PEDIDO". Este botão está associado à *tag* RemotoReq, que por sua vez disparará a confirmação do pedido de usinagem no gerenciador FMC. O *script* responsável pela ação se encontra no Apêndice D desta dissertação. Em conjunto com a confirmação do pedido, o gerenciador remoto também fornece ao gerenciador FMC a informação da identidade do usuário que está requisitando a fabricação. O nome do usuário (*tag* RemotoUser) é atualizado quando o mesmo se autentica no gerenciador remoto por meio de senha. A identidade do usuário será registrada no BD do gerenciador FMC, atrelado ao pedido de fabricação. A figura 5.38 mostra o cadastro de usuários no *software* Elipse SCADA e a autenticação exigida para entrar no aplicativo.

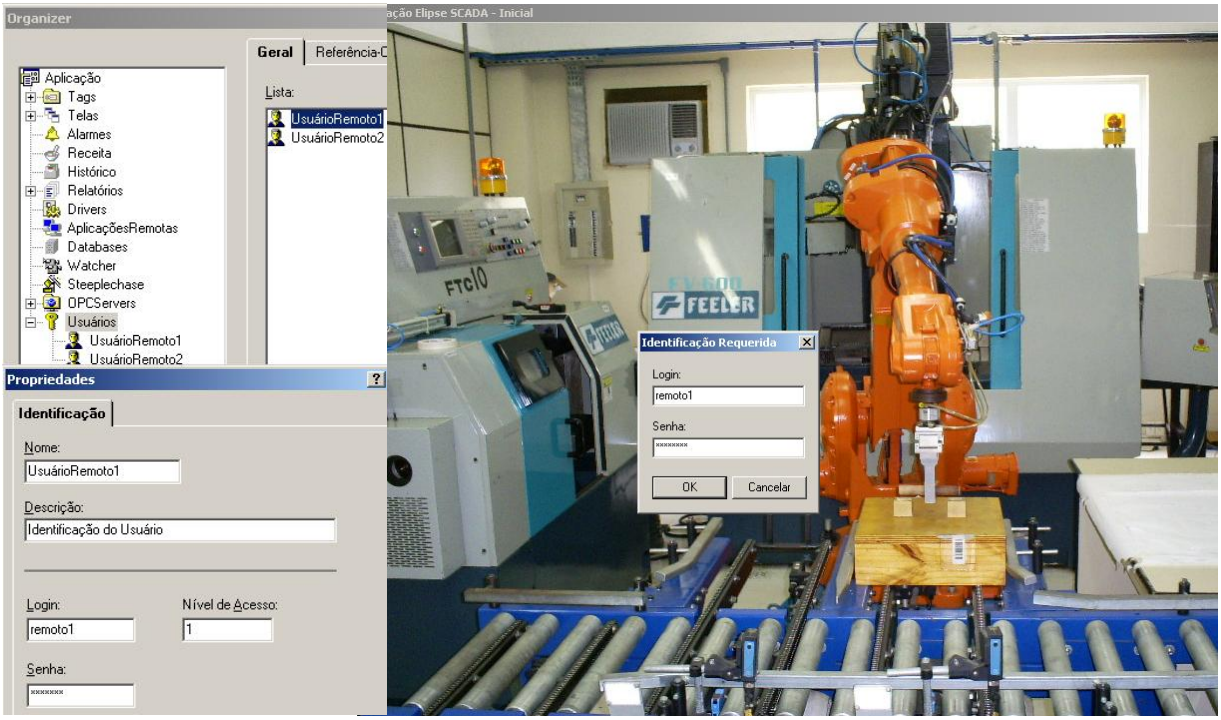


Figura 5.38. Cadastro de usuários e autenticação da aplicação.

O usuário remoto não tem a opção de solicitar prioridade para fabricação de peças, seus pedidos são inseridos na base da pilha de tarefas no BD do gerenciador do armazém. O tempo de conexão do usuário remoto foi limitado a 180 segundos (este tempo é configurável) para que o mesmo não retenha a solicitação de pedidos da tela de "Movimentação", no gerenciador FMC, ou a requisição de outro usuário remoto. Sempre que um usuário remoto entra no sistema, seu *status* ativo é sinalizado no gerenciador FMC. Os estados de movimentação do armazém (entrada, saída e usinagem) também são informados ao usuário remoto, assim como mensagens que confirmam a inserção dos pedidos de fabricação na célula.

5.7 INTEGRAÇÃO GERAL DA CÉLULA

A união dos gerenciadores dos recursos produtivos da célula, com o gerenciador FMC e o gerenciador remoto permite a formação de uma Célula Flexível de Manufatura. A figura 5.39 apresenta um resumo das conexões entre os diferentes gerenciadores, detalhando os recursos de *software* presentes em cada um deles.

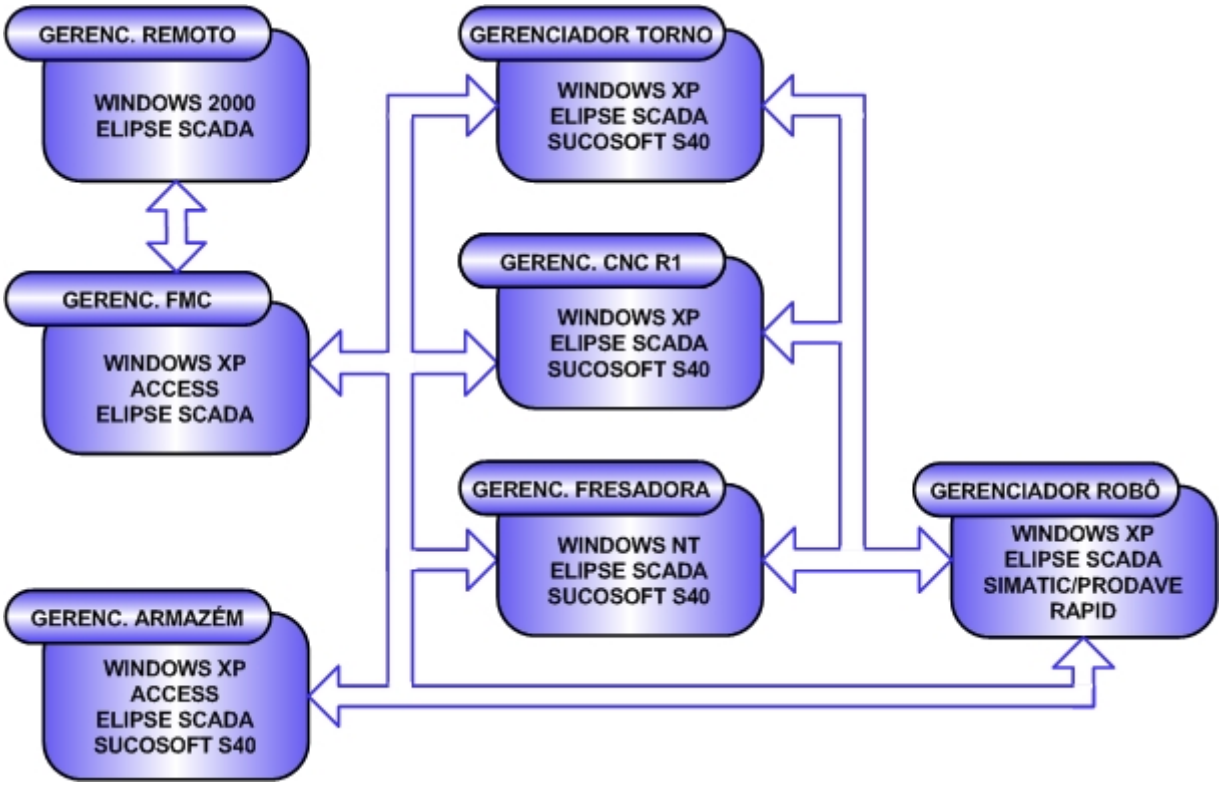


Figura 5.39. Arquitetura das conexões na célula e recursos de *software*.

Todos os gerenciadores estão conectados por meio de uma rede de comunicação local *Ethernet*. Cada gerenciador desempenha o papel de servidor e cliente OPC ao mesmo tempo. Quando um aplicativo fornece informações para outros gerenciadores, ele é um servidor OPC, já quando ele recebe dados de outro gerenciador, comporta-se como cliente OPC.

Diversas versões do sistema operacional Windows foram utilizadas nos gerenciadores (NT, 2000, XP-HOME e XP-SP2) que rodavam em diversos computadores, com *hardwares* variando desde Pentium 200 MMX até P4 2.8 MHz. Todos os computadores contam com placas de rede padrão *Ethernet* e, aqueles que se comunicam com CLPs, com pelo menos uma porta serial RS232; exceção feita ao gerenciador FMC/armazém que possui duas portas seriais (uma utilizada para comunicação com o CLP Moeller *master* e outra com os *scanners* de leitura do código de barra dos *pallets*). O controlador do robô possui um cartão de rede Profibus, modelo DSQC 352, utilizado para comunicação com o CLP Siemens.

5.8 FUNCIONAMENTO DA CÉLULA

Após a integração dos gerenciadores, a célula está apta a produzir as peças previamente cadastradas no banco de dados do gerenciador FMC/armazém. Para iniciar o funcionamento do sistema é necessário que todos os gerenciadores estejam em funcionamento, a rede de comunicação ativa e todos os equipamentos e CLPs da célula com seus programas carregados e ativos.

A interface do usuário com a FMC é o gerenciador do armazém/FMC. Nesta IHM o usuário realiza o cadastro de peças e processos, efetua a entrada e saída de matéria-prima no AS/RS e solicita a fabricação das peças previamente cadastradas no sistema. O acesso do usuário ao gerenciador é realizado por meio de senha, por razões de segurança. O gerenciador FMC/armazém também permite configurar diversas estratégias de movimentação do transelevador, monitorar os principais eventos e estados dos recursos produtivos da célula, emitir relatórios sobre peças produzidas e estocadas e localizar matéria-prima e peças já processadas em estoque no armazém.

Os gerenciadores do robô e dos equipamentos CNC poderiam estar concentrados em um único gerenciador. Neste caso, o CLP utilizado para integrar o gerenciador com os equipamentos deveria contar com portas de entrada e saída suficientes para processar todos os sinais necessários.

O gerenciador remoto é o único dos gerenciadores que não necessita estar ativo durante todo o tempo de funcionamento da célula. Quando o usuário remoto desejar inserir um pedido de fabricação de peças, ele inicia o aplicativo mediante autenticação de senha. O gerenciador localiza e se conecta automaticamente ao servidor OPC FMC/armazém dentro da rede. O tempo de conexão do usuário remoto é restringido a 180 segundos por conexão.

A limitação da família de peças que a célula pode fabricar é dada pela conjugação de diversos fatores, tais como: as dimensões do *pallet* e a capacidade de carga do transelevador; as dimensões e o formato da garra, além da capacidade de carga total do robô (10 kg); as dimensões dos dispositivos de fixação dos equipamentos CNC.

A integração geral dos diversos equipamentos da célula funcionou de maneira correta e conforme previsto na modelagem da célula baseada em RdPI. Para validar a experiência, simulou-se a fabricação de todas as peças cadastradas no BD do gerenciador, o que ocorreu com sucesso. O torno é o único equipamento CNC da célula capacitado a produzir efetivamente, já que o centro de usinagem não conta com nenhum dispositivo de fixação automático e o CNC R1 é um equipamento virtual. Um

único programa de usinagem foi utilizado no torno para a simulação da fabricação das peças. Este programa encontra-se no Anexo em CD-ROM.

A movimentação das peças a partir das mesas de usinagem até os equipamentos CNC é um dos pontos críticos do sistema, pois exige grande precisão posicional das peças no *pallet* e estabilidade da garra do robô na movimentação. Qualquer deslocamento anormal da peça no transporte pode ocasionar a colisão da mesma com o dispositivo de fixação do equipamento CNC. O descarregamento dos equipamentos CNC também requer uma apreciação minuciosa, pois dependendo das dimensões e da geometria final da peça, pode ocorrer que a garra do robô não consiga prender de forma satisfatória a peça processada.

Foram utilizados dois tipos de CLPs (Moeller e Siemens) para efetuar a integração dos gerenciadores com os equipamentos CNC e com o robô. Outras marcas de CLPs podem ser utilizados para cumprir a mesma função, desde que apresentem um número suficiente de I/Os (*Inputs*/*Outputs*).

O robô ABB, modelo IRB 2400/10, de seis graus de liberdade, foi dotado de uma garra para manipulação e transporte de materiais entre o armazém automático e os equipamentos CNC, e vice-versa. Qualquer outro tipo de robô pode ser usado na integração da célula, contanto que seu espaço de trabalho alcance as mesas dos berços de usinagem do armazém e o interior dos equipamentos CNC; que o mesmo esteja dotado de uma ferramenta (garra) apropriada para manipulação e transporte dos materiais utilizados na célula; e que seu controlador possua capacidade de programação e comunicação com o meio externo.

Os equipamentos CNC utilizados, um torno Feeler FTC10 e um centro de usinagem Feeler FV-600, exigiram adaptações de *hardware* e *software* para integração na FMC. As alterações necessárias compreendem a automatização das portas e dos dispositivos de fixação (morsa, pinça, castanha, etc.) e da reserva de um conjunto de memórias internas para indicação de carga dos programas de usinagem. Outras marcas e modelos de CNCs podem ser utilizados na FMC, desde que sejam realizadas as modificações relatadas, se já não forem funcionalidades nativas destes equipamentos.

A inclusão de um novo equipamento CNC na célula seria feita de forma bastante simples. Não é necessário realizar uma nova modelagem da célula, basta construir um novo gerenciador semelhante aos gerenciadores do torno, fresadora e CNC R1. Também devem ser gerados os programas de movimentação do robô para o novo CNC e alterados os *scripts* de programação associados às *tags* OPC.

No caso da inclusão de um sistema de medição, que poderia ser um micrômetro laser ou um sistema de medição por visão, seria necessário definir um conjunto de interfaces para a construção do novo gerenciador. Neste caso também não seria necessário modelar novamente a célula. A movimentação do robô, efetuando o descarregamento dos equipamentos CNC, seria alterada para levar inicialmente a peça até o sistema de medição. Caso a peça fosse aprovada, o robô completaria o movimento de retorno da peça ao *pallet* da mesa de usinagem. Em caso de rejeição, o robô conduziria a peça até um *buffer* de rejeito ou retrabalho.

Entre as propostas inicialmente apresentadas para a construção da célula, previa-se a possibilidade do uso da FMC via Internet. Esta proposta foi temporariamente abandonada em razão de que o módulo de operação do *software* Elipse SCADA via Internet permite apenas monitorar o processo, e não interagir com ele. Um novo gerenciador poderia ser construído, numa eventual continuidade do projeto, com o *software* Elipse E3, que possui um módulo *web* para operações via Internet que interage com o processo, e é compatível com os aplicativos desenvolvidos no Elipse SCADA.

Diversas fotos da FMC, além de um filme que mostra os principais eventos na simulação da fabricação de uma peça, encontram-se no Anexo em CD-ROM desta dissertação.

6. CONCLUSÕES, CONTRIBUIÇÕES E TRABALHOS FUTUROS

Este capítulo apresenta as considerações finais a respeito do trabalho desenvolvido na construção da FMC, as contribuições diretas e indiretas e as sugestões para ampliação e continuidade do projeto de pesquisa.

6.1. CONCLUSÕES

A relevância atual da manufatura flexível pode ser mensurada pela quantidade significativa de trabalhos acadêmicos publicados, na forma de artigos, monografias, dissertações e teses. Muitas instituições de ensino estão instalando laboratórios na área da manufatura integrada por computador, uma proposta que não é recente, porém com implementação complexa e cara. Normalmente são duas as soluções adotadas para a implantação destes laboratórios: a compra de soluções prontas, onde não há transferência de tecnologia para o usuário, com pacotes de *hardware* padronizados e não heterogêneos, juntamente com *softwares* de supervisão e controle fechados, verdadeiras "caixas pretas". O segundo tipo de solução adotada é a integração de diferentes equipamentos (manuseio, transporte e processamento) fornecido por diferentes fabricantes, para a formação de uma FMC ou FMS. Neste caso, todo o processo de integração deve ser construído, desde a modelagem até o sistema de controle, passando pela programação de CNCs, CLPs, robôs, alterações de *hardware* dos equipamentos e estabelecimento de padrões de comunicação. O trabalho desta dissertação abordou esta segunda proposta para a construção de uma FMC.

A definição de um conjunto mínimo de interfaces para cada equipamento da célula, aliado à definição de um sistema de controle heterárquico, determinou o conjunto de mensagens necessárias para a integração dos diferentes equipamentos. A modelagem da célula em rede de Petri Interpretada permitiu traduzir o comportamento dinâmico da FMC diretamente para os códigos do *software* de supervisão e controle. O disparo de cada transição na modelagem, associado a um conjunto de condições, também foi diretamente transportado para a execução de *scripts* de programação no *software* Elipse SCADA. As ações decorrentes do disparo das transições, definidas na modelagem, foram transcritas de forma integral para os códigos de execução dos *scripts*.

Todos os procedimentos metodológicos e técnicos efetuados para realizar a integração física e lógica dos equipamentos do laboratório da SOCIESC configuram uma diretriz, um conjunto de

instruções e procedimentos para a construção de uma FMC. Há na literatura técnica uma grande deficiência no sentido de apresentar soluções detalhadas para integração de equipamentos heterogêneos. Os fabricantes de equipamentos industriais e as empresas de automação que detêm o domínio destas tecnologias guardam a "sete chaves" seus segredos industriais, pois eles representam um grande diferencial competitivo que reverte em benefício financeiro próprio. Durante o desenvolvimento do projeto de pesquisa, várias foram as vezes em que foi necessário contatar os fabricantes dos equipamentos para dirimir dúvidas, ou complementar alguma informação técnica essencial para o projeto. Raras foram as vezes em que os fabricantes forneceram algum tipo de informação válida, antes, o comportamento comum foi tentar vender a informação solicitada. Os manuais técnicos que acompanham os equipamentos também apresentam grandes deficiências, pois via de regra eles podem ser considerados apenas como manuais operacionais. Como exemplo deste caso, pode-se mencionar o caso do torno CNC da Feeler que utiliza internamente um conjunto de CLP/CNC da Mitsubishi. Os manuais que acompanham o torno não contêm nenhuma informação técnica mais profunda a respeito do CLP/CNC da Mitsubishi. Neste caso foi necessário adquirir os manuais diretamente do representante da Mitsubishi, o que encareceu e atrasou o andamento do projeto de pesquisa.

Em razão das dificuldades do acesso às informações técnicas sobre equipamentos, e sendo fiel à pergunta de pesquisa inicialmente formulada (Como integrar de forma física e lógica os diversos equipamentos existentes no laboratório da SOCIESC, utilizando uma filosofia modular, aberta e expansível...?), esta dissertação também apresenta de forma detalhada todos os aspectos de engenharia envolvidos no projeto de pesquisa. Estes aspectos técnicos se estendem a detalhes de *hardware* dos equipamentos, programas de CLPs, programas de movimentação do robô, parametrização de *drivers*, *scripts* de programação do supervisório SCADA e configuração gerais de *software* e de comunicação. O anexo em CD-ROM traz informações técnicas adicionais, na forma de tabelas, listas, figuras, fluxogramas e diagramas entidade-relacionamento sobre o armazém automático, o robô, os equipamentos CNC da Feeler e os CLPs Moeller e Siemens. Sem estas informações técnicas detalhadas, a aplicação prática das diretrizes propostas pode ser comprometida, justamente pela dificuldade de obtenção de dados técnicos necessários à integração dos diferentes equipamentos.

A integração dos equipamentos no laboratório da SOCIESC permite que o mesmo seja utilizado em diversas disciplinas dos cursos de graduação e pós-graduação. A filosofia utilizada para a construção do sistema de controle, através do uso de um supervisório SCADA, e a comunicação entre os diversos gerenciadores sendo realizada via OPC, resulta em um sistema aberto, modular e

expansível. Melhorias e modificações poderão ser realizadas no futuro por trabalhos de conclusão de curso de graduação e dissertações de Mestrado.

6.2. CONTRIBUIÇÕES

A maior contribuição deste projeto de pesquisa é oferecer uma diretriz para construção de uma FMC, a partir da integração física e lógica de equipamentos heterogêneos. O roteiro desta proposta compreende a modelagem do sistema utilizando Redes de Petri Interpretadas, a determinação de um conjunto de interfaces para cada equipamento, a construção de aplicativos gerenciadores baseados em um supervisório SCADA, a integração dos gerenciadores com os equipamentos CNC e o robô, utilizando CLPs, e a integração dos gerenciadores por meio do padrão de comunicação industrial OPC. Desmembrando as contribuições em tópicos principais, pode-se mencionar:

- A efetiva construção de uma FMC, a partir de um conjunto de recursos produtivos que inicialmente não possuía nenhum tipo de integração.
- Construção dos gerenciadores dos equipamentos CNC: foram construídos três gerenciadores (torno, fresadora e CNC R1), que cumprem a função de controlar e integrar os CNCs à célula e servir de interface gráfica com o usuário, fornecendo informações a respeito dos principais estados destes equipamentos;
- Construção do gerenciador do robô: controla e integra o robô na FMC, além de fornecer uma interface gráfica com o usuário, contendo informações sobre a garra, a movimentação e os estados do robô.
- Construção dos gerenciadores FMC e remoto: O gerenciador da FMC concentra as informações principais de todos os equipamentos da célula, gerencia a codificação das peças e processos no banco de dados, além de servir como interface com o usuário da célula, permitindo que sejam inseridos pedidos de fabricação. O gerenciador remoto permite que um usuário, que se encontra dentro da rede, também possa inserir seus pedidos de fabricação.
- Alteração da IHM do armazém automático: o aplicativo original foi substancialmente modificado para permitir a integração do armazém AS/RS. Alguns defeitos crônicos do armazém, decorrentes de problemas entre os códigos originais do supervisório e o programa

do CLP *master*, foram minimizados ou resolvidos, mediante a construção do gerenciador do armazém.

- Integração entre o gerenciador e o torno CNC – A integração do gerenciador com o torno demandou a utilização de um CLP Moeller, que estabeleceu um meio de comunicação com o CLP do robô, um Mitsubishi Meldas 50. Novos programas de CLP foram escritos, enquanto outros foram modificados. Alterações de *hardware* também foram introduzidas, para automatizar algumas funções do torno e permitir a integração do mesmo com o gerenciador.
- Integração entre o gerenciador e o robô: O gerenciador se integra ao robô através de um CLP Siemens. Vários programas de movimentação de peças foram escritos utilizando a linguagem RAPID. A conexão entre o CLP Siemens e as I/Os digitais do robô utiliza o padrão Profibus, fato que abre a possibilidade futura de conexão direta entre o gerenciador e o robô utilizando este padrão, já que o laboratório conta com um computador equipado com uma placa de rede padrão Profibus.
- Geração de bibliografia para continuidade do projeto: Todos os procedimentos efetuados para a integração da célula são descritos em detalhes, desde o desenvolvimento dos gerenciadores, as *tags*, configurações, parâmetros e *scripts*; além dos programas dos CLPs e do robô. Os diagramas elétricos das modificações no torno e as configurações da comunicação OPC também foram incluídos nesta dissertação.

6.3. TRABALHOS FUTUROS

A validação do problema de pesquisa inicialmente formulado foi efetuada com êxito, resultando em uma proposta de um conjunto de procedimentos e instruções para a construção de uma FMC a partir de um sistema de armazenamento automático, um robô industrial e um grupo de máquinas CNC. Porém, pode-se afirmar que, diante do potencial do laboratório e da importância do tema, apenas o primeiro passo foi dado. Há muitos aspectos que podem ser melhorados, e outros que devem ser construídos, para avançar no sentido apontado pela filosofia CIM. Algumas considerações sobre possíveis trabalhos futuros, são:

- Implementar uma morsa hidráulica no centro de usinagem CNC, para possibilitar sua total integração na célula. Atualmente a utilização da fresadora é limitada pela impossibilidade de

fixar automaticamente as peças fornecidas pelo robô. Por enquanto, a usinagem destas peças pode ser apenas simulada;

- Um novo gerenciador FMC deve ser construído, utilizando o supervisório Elipse E3, que fornece a possibilidade de incorporar todos os gerenciadores anteriormente desenvolvidos no Elipse SCADA, além de possibilitar e a operação remota da FMC via Internet. O *software* Elipse SCADA possui um módulo de operação através da Internet, porém este não permite interagir com o processo, mas apenas monitorá-lo. O novo gerenciador da FMC deverá ser hierarquicamente superior aos demais gerenciadores, modificando a hierarquia do sistema;
- O desenho atual do *pallet* permite o transporte de apenas uma peça. Um novo modelo deve ser desenvolvido, com maior estabilidade dimensional e capacidade de carga de várias peças. Esta mudança demandará alterações no banco de dados do gerenciador do armazém;
- Construção de uma garra angular para o robô. A garra utilizada atualmente apresenta problemas para transportar peças cilíndricas com estabilidade. Já existe um projeto (mostrado no Anexo) para a construção deste novo equipamento;
- Implantação de um sistema de medição das peças produzidas, para controle de qualidade. Este sistema pode ser composto por um micrômetro laser ou um sistema de reconhecimento por visão;
- Transferência de programas de usinagem diretamente do gerenciador para o torno e para o centro de usinagem CNC. Atualmente os programas de usinagem são previamente carregados na memória dos CNCs. Com a transferência direta, a célula ganha em flexibilidade, pois novas peças poderão ser inseridas à família, sem a necessidade de reprogramar o gerenciador e instalar antecipadamente os programas nos CNCs. Este item foi testado, porém não chegou a ser implementado por restrições temporais. Módulos de comunicação, associados ao gerenciador dos equipamentos CNC foram desenvolvidos e testados com sucesso, porém este procedimento ainda exige a ação do operador da máquina CNC. Modificações de *hardware* se fazem necessárias no torno e no centro de usinagem para automatizar o procedimento de transferência de programas de usinagem a partir dos gerenciadores.

REFERÊNCIAS BIBLIOGRÁFICAS

ABB. Manuais fornecidos pelo fabricante:
-Manual do Produto IRB-2400 ABB;
-RAP Protocol Specification TRP-1 (RPC,TCP/IP,SLIP).

AGUIAR, Renato Armani. **Células de Manufatura: uma abordagem conceitual.** Monografia, Universidade Federal de Ouro Preto, 2003.

ALVES, Raul; NORMEY-RICO, Júlio E; MERINO, Alejandro; PRADA, César de. **Un SCADA via OPC aplicado a una planta piloto**. 2º Congresso Brasileiro de P&D em Petróleo & Gás. Rio de Janeiro, 2003.

BARROS, Marcelo Ramos de Albuquerque. **Sistema Supervisório – Ferramenta vertical para gerenciamento flexível de dados de produção**. In: Controle & Instrumentação, Outubro 2005, p. 107-111.

BATOCCHIO, Antônio; FIORONI, Marcelo Moretti. **Arquitetura dos Sistemas de Manufatura**. Artigo disponível na Internet em:< http://www.fem.unicamp.br/ ~defhp/gphms/artigo10.doc.> Acesso em 05/10/2006>.

BOARETTO, Neury; SANZOVO, Nádia; SCANDELARI, Luciano. **Implantação de um laboratório de automação de manufatura na unidade Pato Branco do CEFET-PR**. Anais do COBENGE, 2004.

BRUNSTEIN, Israel; ESTRELA, George Queiroga; SEVERIANO FILHO, Cosmo. **Análise da medição da produtividade na indústria de curtume do Brasil**. Anais do VIII Congreso Internacional de Costos. Punta Del Este, Asociación Uruguaya de Costos, 2003.

CARDOSO, Janette; VALETTE, R. **Redes de Petri**. Editora da UFSC, Florianópolis, 1997, 212 p.

CECCONELLO, Ivandro. **Adequação de um sistema de administração da produção à estratégia organizacional**. Dissertação de Mestrado, UFSC, Florianópolis, 2002.

CERVO, Amado Luiz; BERVIAN, Pedro Alcino. **Metodologia científica**. 3ª edição, McGraw-Hill do Brasil, São Paulo, 1983.

CHASE, Richard B.; AQUILANO, Nicholas J.; JACOBS, F. Robert. **Administración de producción y operaciones. Manufactura y servicios.** Irwin Mc Graw-Hill, Madrid, 2000.

CHAVES, Charles da Silva; ALBANO, Sidnei José. **Implantação de comunicação via rede Profibus entre PC e robô**. Monografia. IST, 2006.

COINBRA, Bernhar Gobbi Rocha; BARBOSA, Henrique Damiani Santana; SOUZA JÚNIOR, José Leonardo Neves de. **Construção de um sistema de produção flexível**. Monografia.Universidade Católica de Goiás, 2004.

CORRÊA, Henrique Luiz; SLACK, Nigel D. C. **Flexibilidade estratégica na manufatura**. Anais do ENEGEP, 1994.

COSTA, Luis Sergio Salles; CAULLIRAUX, Heitor M. **Manufatura integrada por computador: estratégia, organização, tecnologia e recursos humanos**. Editora Campus, SENAI, COPPE/UFRJ, Rio de Janeiro, 1995, 450p.

CURY, José Eduardo Ribeiro. **Teoria de Controle Supervisório de Sistemas a Eventos Discretos**. V Simpósio Brasileiro de Automação Inteligente. Apostila, DAS, UFSC, 2001.

DANEELS, Alex; SALTER, Wayne. **What is SCADA?** CERN - European Organization for Nuclear Research, CNL-2000-003, Vol. XXXV, issue nº 3.

DAYTON-KNIGHT Ltd. **SCADA Explained**. Artigo disponível em <http://www.dayton-knight.com/Projects/SCADA/scada_explained.htm>. Acesso em 09/02/2006.

DE NEGRI, Victor Juliano. **Introdução aos Sistemas para Automação e Controle Industrial**. Apostila, LASHIP, EMC, UFSC, Florianópolis, 2004.

ELIPSE SCADA. **Manual do Usuário**, HMI/Scada Software, Versão 2.28, 2006, 351 p.

FAYET, Eduardo Alves. **Sistemas Logísticos Integrados: um rol de critérios para análise**. Dissertação de mestrado, UFSC, Florianópolis, 2002.

FEELER. **Manual elétrico para CNC com controle Meldas M50 para torno FTC-10**. Manual fornecido pelo fabricante.

FERREIRA, João Carlos Espíndola. **Integração de uma Célula Flexível de Manufatura e seu uso na Fabricação Local e Remota de Peças**. Florianópolis, 2001. Projeto Integrado de Pesquisa – Departamento de Engenharia Mecânica da UFSC.

FERREIRA, João Carlos Espíndola. **Sistemas Integrados de Manufatura**. Apostila, GRIMA/GRUCON/UFSC, Florianópolis, 1998.

FERREIRA, João Carlos Espíndola. **Planejamento do processo assistido por computador (CAPP)**. Apostila, GRIMA/GRUCON/UFSC, Florianópolis, 2005.

FERREIRA, João Carlos Espíndola; SANTOS, Marcelo Teixeira dos; SCHIRMER, Léo; et al. **Um Método para a Fabricação de Peças à Distância via Internet num Sistema Flexível de Manufatura - FMS**. In: revista do IST, Joinville, ano 3, nº04, p. 25-30. Outubro de 2003 (a).

FERREIRA, João Carlos Espíndola; SANTOS, Marcelo Teixeira dos; ÁLVARES, Alberto José; et al**. A Procedure For Integrating Automated Equipment In A Flexible Manufacturing System And Their Use For The Remote Manufacture Of Parts Through The Internet**. Anais do COBEM, 2003 (b).

FIELDBUS Foundation. Disponível em <http://fieldbus.org>. Acesso em 05/01/2006.

FRIEDRICH, Luis Fernando. **Uma Abordagem Distribuída no Desenvolvimento e Implementação do Software de Controle de Chão-de-Fábrica em Sistemas de Manufatura Celular**. Tese de doutorado, UFSC, 1996.

FONSECA, Marcos de Oliveira. **Comunicação OPC – Uma abordagem prática**. Anais do VI Seminário de Automação de Processos, Vitória, ES, 2002.

GAIDZINSKI, Vladimir Hartenias. **A Tecnologia da Informação no chão de fábrica: as novas ferramentas e a gestão integrada da informação**. Dissertação de mestrado, UFSC, 2003. 153 p.

GÓMEZ, Arthur Tórgo; LORENA, Luiz Antonio N. **Modelagem de Sistemas Flexíveis de Manufatura considerando restrições temporais e a capacidade do magazine**. Artigo disponível na Internet em: <http://www.lac.inpe.br/~lorena/asmf.ps>. Acesso em 10/09/2006.

GUSTIN, Gladys Deifan Bastidas. **Aplicação de Redes de Petri Interpretadas na Modelagem de Sistemas de Elevadores em Edifícios Inteligentes.** Dissertação de Mestrado, Escola Politécnica da Universidade de São Paulo, 1999.

HAAS, Fernando; DETTRUZ, Giovanni. **Integração de um torno CNC para a formação de uma Célula Flexível de Manufatura.** Monografia. IST, 2004.

HILLER, Diogo. **Configurando o Windows XP Service Pack 2 para aplicações Elipse**. Artigo atualizado em 08/09/2004, disponível em <http://www.elipse.com.br/elipse /artigos.aspx>. Acesso em 23/01/2006.

JIMÉNEZ, E; RECLADE, L; SILVA, M. **Visión comparativa entre redes de Petri continuas y diagramas de Forrester**. XXIII Jornadas de Automática, Santa Cruz de Tenerife, España, 2002.

KLUG, Robson Roberto. **Sistema Flexível de Manufatura**. Monografia. IST, 2000.

LEPIKSON, Herman Augusto. **Sistemas Integrados de Manufatura**. In: Tecnologias avançadas de manufatura – Coleção fábrica do milênio, v. 1. Editora Novos Talentos, 2005, p. 13-34.

LINO, Rui; GOMES, Luis. **Detecção de falhas de sensores em sistemas de automação utilizando Redes de Petri**. 3ª Jornada de Engenharia de Electrônica e Telecomunicações e de Computadores, Lisboa, 2005

LORINI, Flávio José. **Tecnologia de Grupo e Organizações da Manufatura**. Editora da UFSC, Florianópolis, 1993.

MAcCARTHY, B.L. e LIU, J. **A new classification scheme for flexible manufacturing Systems**, International Journal of Production Research, 31(2), 299-309, 1993.

MACIEL, Paulo Henrique Soares. **Elipse SCADA como cliente OPC**. Artigo atualizado em 24/01/2005, disponível em <http://www.elipse.com.br/elipse/artigos.aspx>. Acesso em 25/01/2006.

MALER, Oded. **On the programming industrial computers**. Artigo publicado em 04/06/1999 e disponível na Internet em <http:// www-verimag.imag.fr/VHS/IP/iec1131.ps>. Acesso em 10/10/2006.

MARRANGHELLO, Norian. **Redes de Petri: Conceitos e aplicações.** Apostila do DCCE/IBILCE/UNESP, Março de 2005.

MARTINS, Petrônio Garcia; LAUGENI, Fernando Piero. **Administração da produção**. Editora Saraiva, São Paulo, 1998.

MITSUBISHI ELECTRIC. Manuais fornecidos pelo fabricante.
- Alarm/Parameter Manual - Meldas 60/60S series;
- Custom Release (APLC) Programmig Manual - Meldas 60 series;
- DDB Interface Manual – CNC Meldas 60/60S series;
- Operation Manual - Meldas 60/60S series;
- PLC Onboard Instruction manual - Meldas 60/60S series;
- PLC Program Development Manual (For personal computer) - Meldas 60/60S series;
- PLC Programming Manual (Ladder section) - Meldas 60/60S series;
- Specifications Manual - Meldas 50 M50;
- Specifications Manual - Meldas 60/60S séries.

MOELLER. **Complemento AWA 27-1623**. Manual fornecido pelo fabricante.

MUŠIČ, G; MATKO, D. **Petri net based supervisory control of flexible batch plants**. 8th IFAC/IFORS/IMACS/IFIP Symposium on Large Scale System, Greece, Vol.2, pp. 989-994, 1998.

NASCIMENTO, Gabriel Lenna do**. O padrão OPC e o desenvolvimento de SCADAS de código fonte aberto**. In: Controle & Automação, Maio 2005, p. 127-129.

OPC Foundation. **OPC Overview**. Version 1.0, 1998.

ORTIZ, Luis E. C.; LIMA, Antonio M. A.; MUZZI JUNIOR, José J. **Alternativas brasileiras para viabilização de FMS – Tendências e problemas**. In: Revista A&C, Março 1991, p. 15-20 e Abril 1991, p.23-26.

PELINI, Ezequiel. **Comunicação e controle de um centro de usinagem CNC visando a implementação de um sistema integrado de manufatura.** Relatório de estágio. Universidade de Caxias do Sul, 2001.

PELS, H.J; WORTMANN, J.C; ZWEGERS, A.J.R. **Flexibility in manufacturing: An architecture point of view**. In: Computer in industry 33, p. 271-283, 1997.

PINA, Israel Benítez; VILCHEZ, José Ruben Sicchar; SANSONE, José Luiz; DEL RIO, Daniel Guzman. **Modelagem de automação de Sistemas Flexíveis de Manufatura com PN GHENESYS IEC61131 compatível**. Anais do Congresso Brasileiro de Automática – CBA 2006.

PIRES, Paulo Sérgio Motta; OLIVEIRA, Luiz Affonso H. Guedes de; BARROS, Diogo Nascimento. **Aspectos de segurança em sistemas SCADA – Uma visão geral**. 4º Congresso Internacional de Automação, Sistemas e Instrumentação. In: Controle & Instrumentação, Maio de 2005, p.112-119.

PROFIBUS. **Descrição Técnica do Process FieldBus**. São Paulo, Associação Profibus Brasil, Outubro 2000.

QUEIROZ, Max H. de; CURY, José Eduardo Ribeiro. **Controle Supervisório modular de sistemas de manufatura**. In: Controle & Automação vol.13, n2, p. 123-133, Maio/Agosto 2002.

REMBOLD, Ulrich; NAJI, Bartholomew O; e STORR, Alfred. **Computer Integrated Manufacturing and Engineering**. Addison-Wesley Publishing Company, 1993.

RIBEIRO, Hélder; ELVAS, Ricardo. **Supervisão de Sistemas de Automação com aplicação a uma Célula de Fabricação Flexível**. Monografia. Universidade Técnica de Lisboa, 2004.

ROHDE, Leonardo Rosa; BORENSTEIN, Denis. **Representação em espaço de estados para a flexibilidade de roteamento**. In: Gestão & Produção, v.11, n2, p. 251-261, Maio/Agosto de 2004.

SALVADOR, Marcelo Barbosa. **Tecnologias para criação de sistemas híbridos a partir de sistemas SCADA e CLPs**. In: Controle & Instrumentação, Junho 2005, p. 92-96.

SANCHES, Mar Stella. **CIM – Manufactura integrada por computador**. Disponível em <http://www.elprisma.com/apuntes/ingenieria_industrial/cimmanufacturaintegradaporcomputadora/default5.asp>. Capítulo 4, p. 38-47. Acesso em 10/09/2006.

SANTOS, Hugo Gaspar. **Integração de um Armazém Automático, um Torno CNC e um Robô Industrial, para a Formação de uma Célula Flexível de Manufatura**. Monografia, IST, 2003.

SAVARIS, Charles Edsom; POSSAMAI, Osmar. **Modelo para identificação dos recursos que limitam a flexibilidade da manufatura**. Anais do 3º COBEF, Joinville, 2005.

SEGURA, Ricardo Bonfim; MIELLI, Fábio Marcelus. **O Ethernet TCP/IP Modbus**. In: Controle & Instrumentação, Outubro 2005, p. 117-120.

SEIXAS FILHO, Constantino. **A produção em foco**. In: Scantech News, Rio de Janeiro, Setembro 1999, p. 26-30.

SEVERIANO FILHO, Cosmo. **O Enfoque Vetorial da Produtividade em um Sistema de Avaliação para a Manufatura Avançada na Indústria de Alimentos**. Tese de Doutorado, Florianópolis, UFSC, 1995.

SCHEFFER. **Sistema Tupi Transelevador**. Manual Elétrico e Manual do Sistema.

SIEMENS. **Simatic Net DPC1 Programming Interface**. Manual fornecido pelo fabricante.

SILVA, Ana Paula da; SALVADOR, Marcelo. **O que são sistemas Supervisórios?** Artigo atualizado em 20/12/2005, disponível em: <http://www.elipse.com.br/elipse/artigos.aspx>. Acesso em 17/01/2006.

SILVA, Edna Lúcia da; MENEZES, Estera Muszkat. **Metodologia da pesquisa e elaboração da dissertação**. 4ª edição revisada e atualizada, Florianópolis, UFSC, 2005.

SLACK, Nigel; CHAMBERS, Stuart; JOHNSTON, Robert. **Administração da Produção.** Tradução: Maria Teresa Correa de Oliveira, Fábio Alher. 2º ed. São Paulo: Atlas, 2002.

TEIXEIRA, Evandro Leonardo S.; CANO, Carlos E. Villanueva; ÁLVARES, Alberto J. **Modeling and implementation of a flexible manufacturing cell (FMC)**. Anais do COBEM, Ouro Preto, 2005.

TEIXEIRA, Evandro Leonardo Silva. **Desenvolvimento da unidade de gerenciamento de uma célula flexível de manufatura integrada a um sistema CAD/CAPP/CAM.** Dissertação de Mestrado, 2006 – Universidade de Brasília, 176p.

TENG, Sheng-Hsien; ZHANG, Jie. **A Petri net-based decomposition approach in modeling of manufacturing systems**. International Journal of Production Research, 31(6), 1423-1439, 1993.

TORRICO, César R. C; CURY, José E. R. **Controle supervisório hierárquico modular por agregação de estados**. In: Controle & Automação, Vol 15, número 3, p. 291-300, 2004.

VARELA, César E.; ACOSTA, José E.; SASTRÓN, Francisco. **Modelo orientado a objetos del género equipo para su integración en sistemas de manufactura.** Instituto Tecnológico de Chihuahua, México. In: Electro 2001, p. 315-320.

VIEIRA, Guilherme Ernani. **Integração, Gerenciamento e Implantação Didática de Células Flexíveis de Manufatura**. Dissertação de Mestrado, Florianópolis, 1996, 179 p.

WONHAM, W.; RAMADGE, P. **The control of discrete event system**. Proceeding of the IEEE 77(1):81-98, 1989.

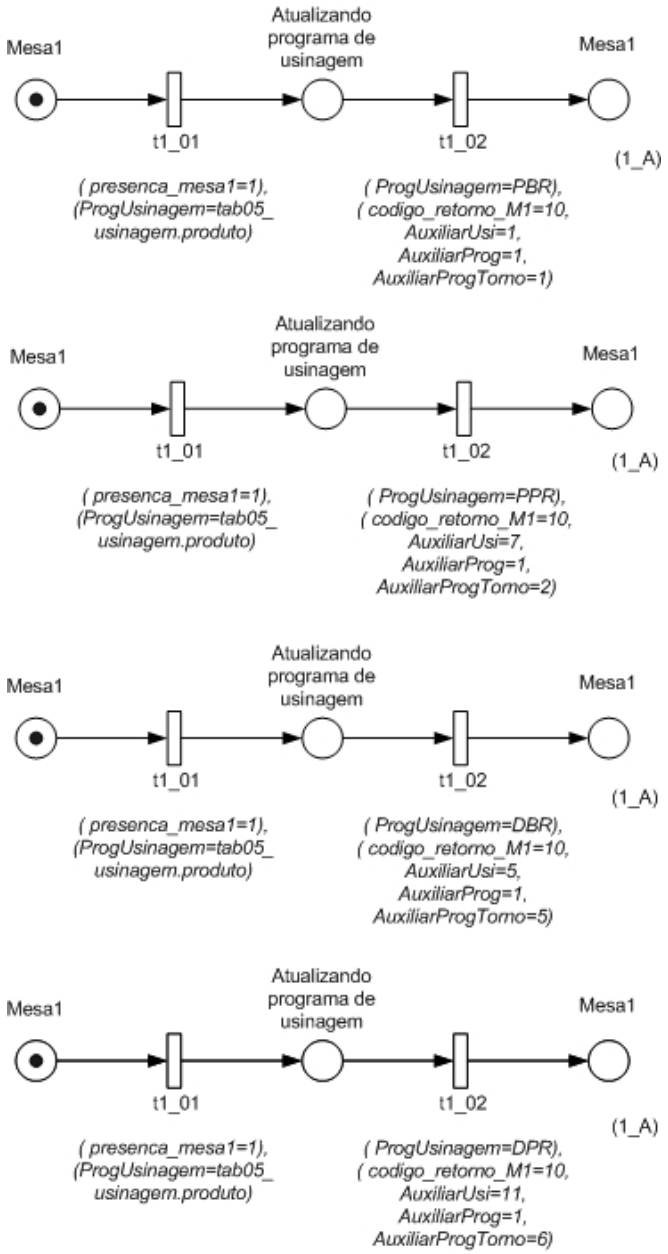
ZHOU, MengChu; VENKATESH, Kurapati. **Modeling, simulation, and control of flexible manufacturing systems**. World Scientific Printers, 2000, 409 p.

APÊNDICE

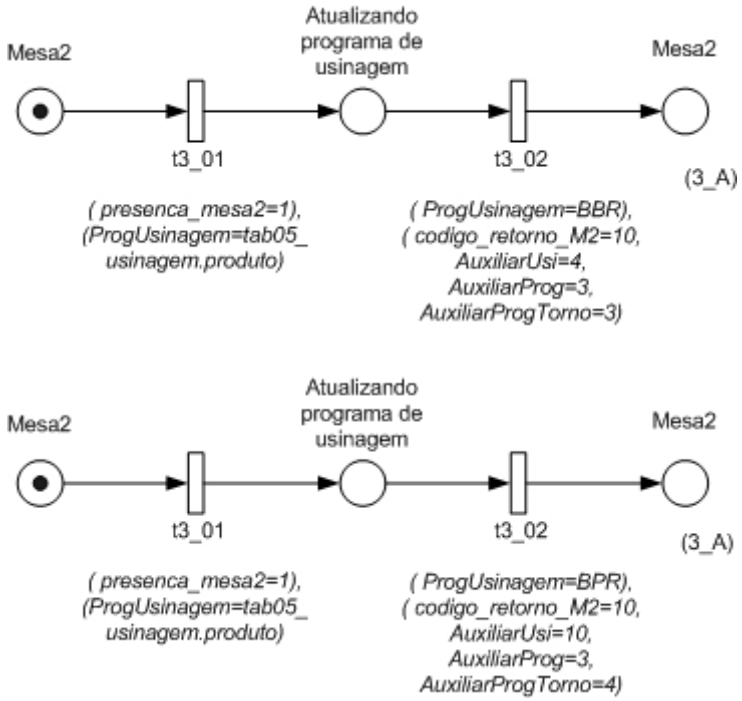
APÊNDICE A - RdPI DA CÉLULA FLEXÍVEL DE MANUFATURA

O Apêndice A apresenta as Redes de Petri Interpretadas utilizadas na modelagem da Célula Flexível de Manufatura. As variáveis que representam as condições de disparo de cada transição, assim como as ações correspondentes, são apresentadas na forma literal adotada na construção dos códigos do controle supervisório.

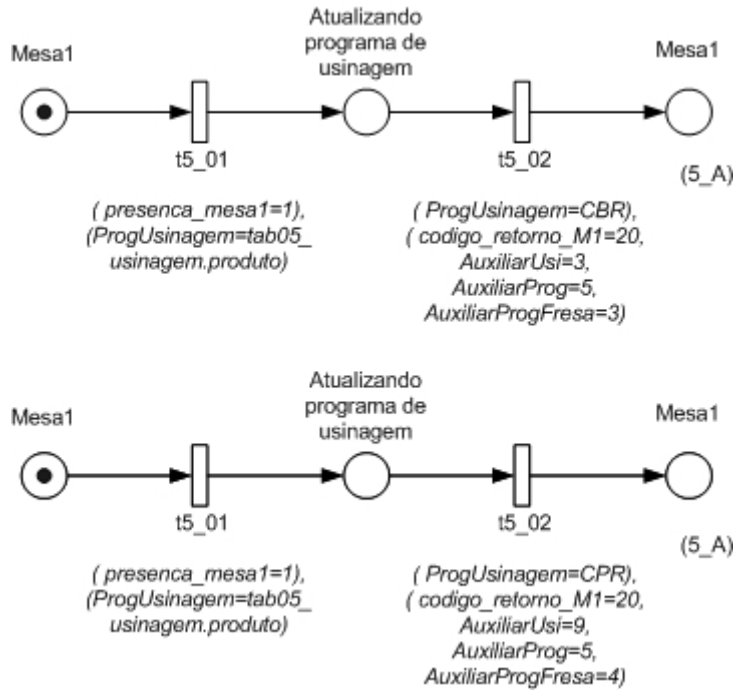
A1. Redes de Petri Interpretadas da integração da mesa1 do AS/RS com o torno.



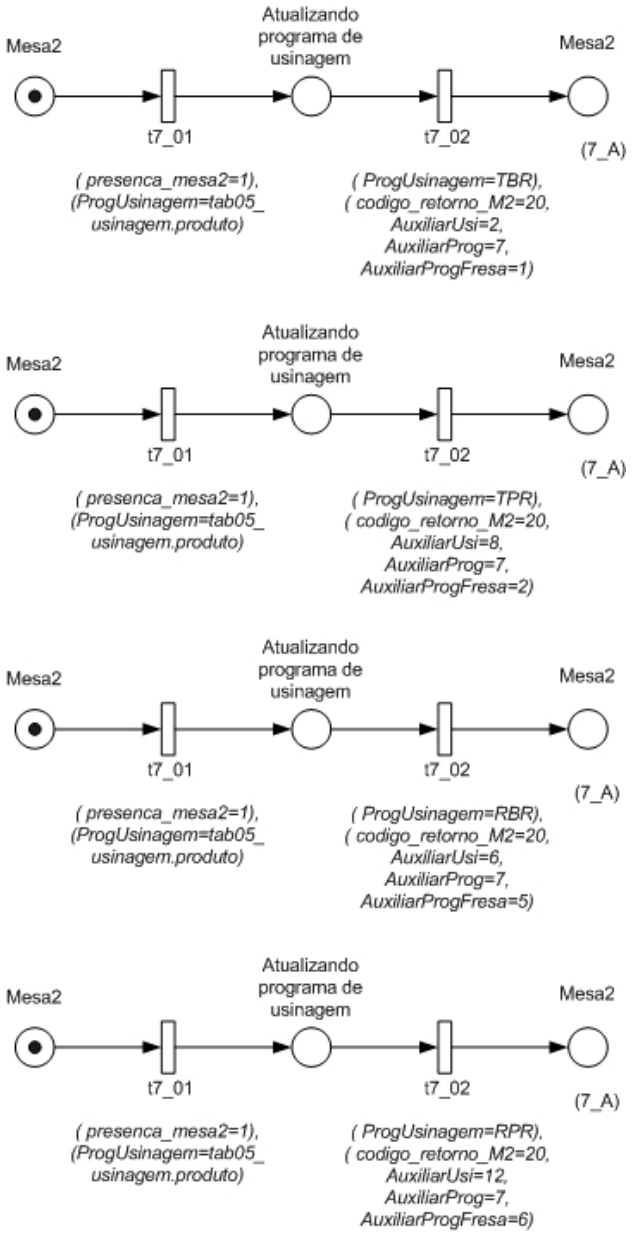
A2. Redes de Petri Interpretadas da integração da mesa2 do AS/RS com o torno



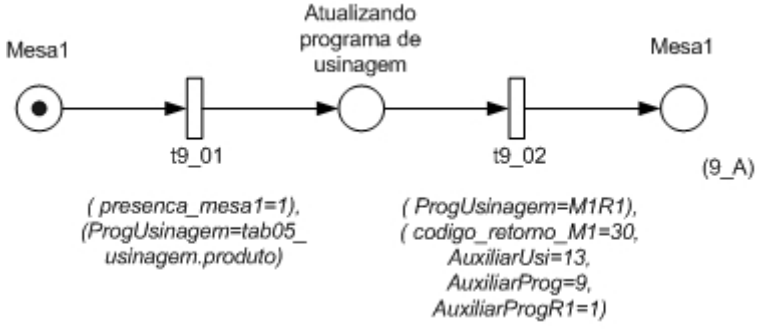
A3. Redes de Petri Interpretadas da integração da mesa1 do AS/RS com a fresadora



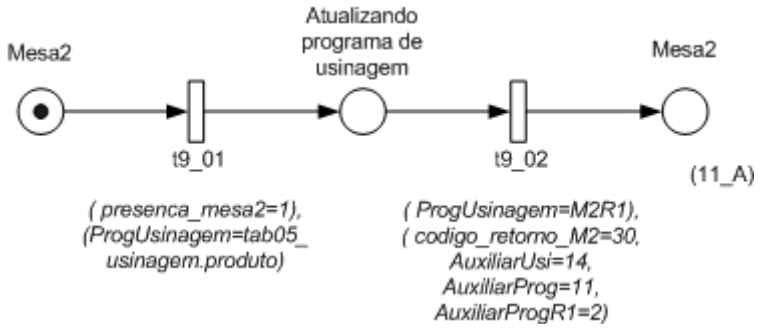
A4. Redes de Petri Interpretadas da integração da mesa2 do AS/RS com a fresadora



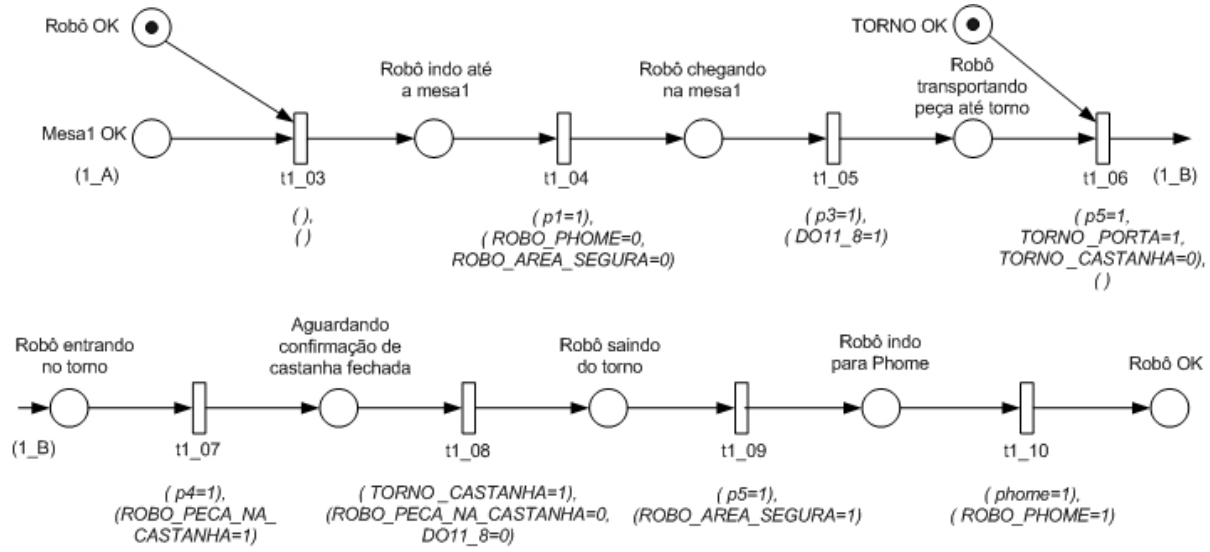
A5. Redes de Petri Interpretadas da integração da mesa1 do AS/RS com o CNC R1

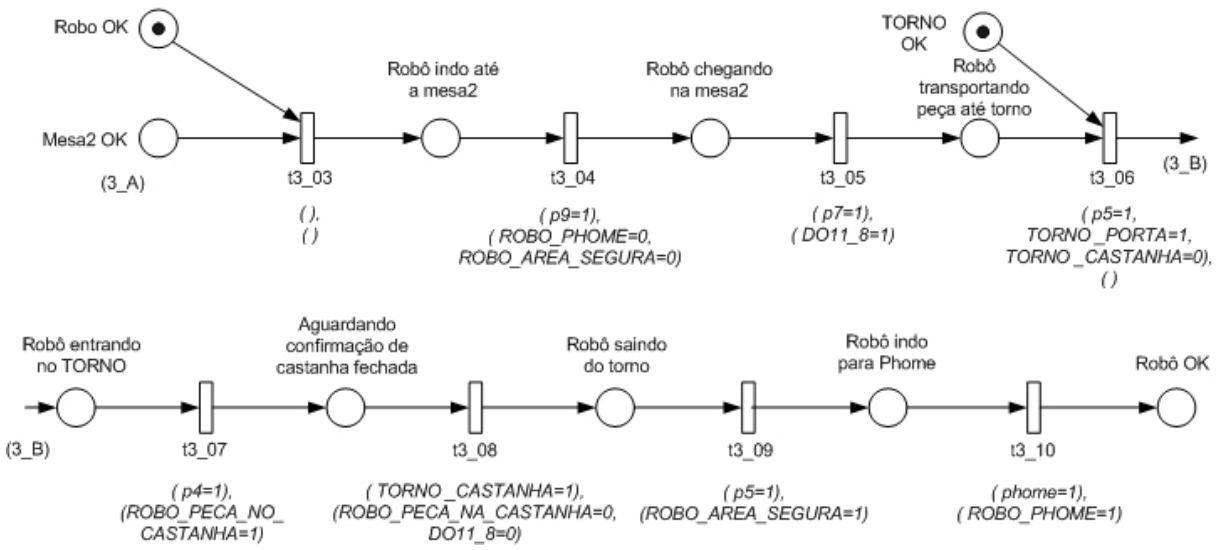


A6. Redes de Petri Interpretadas da integração da mesa2 do AS/RS com o CNC R1

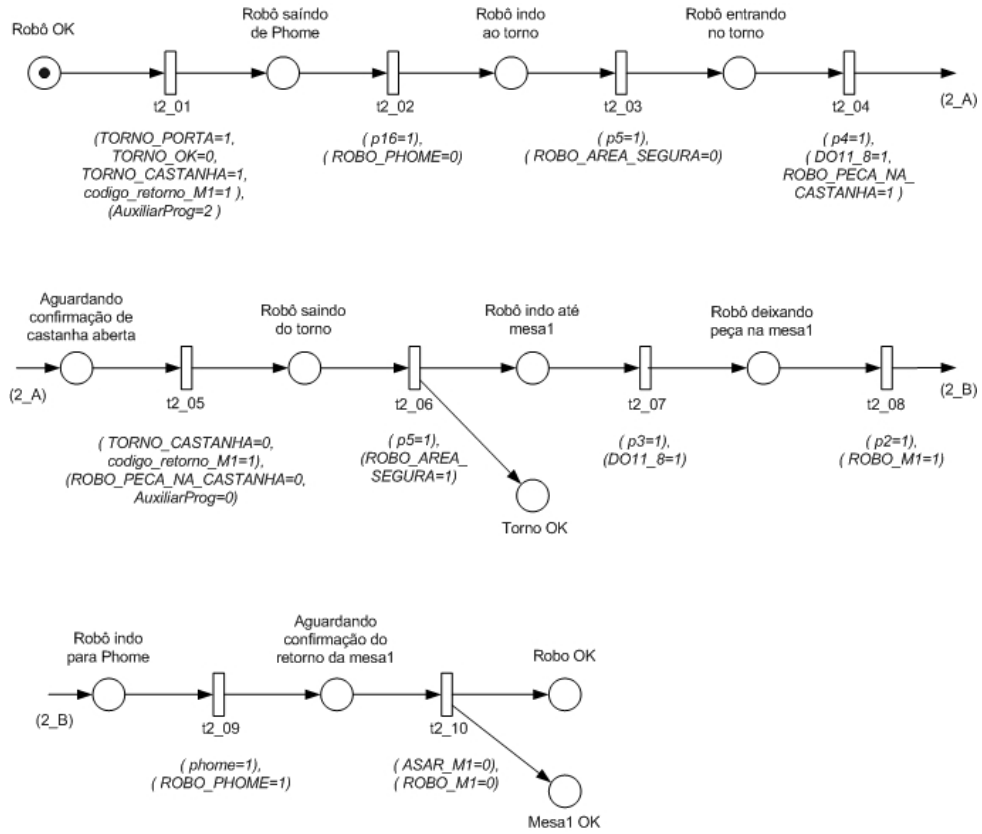


A7. Redes de Petri Interpretadas da integração das mesas do AS/RS com o torno

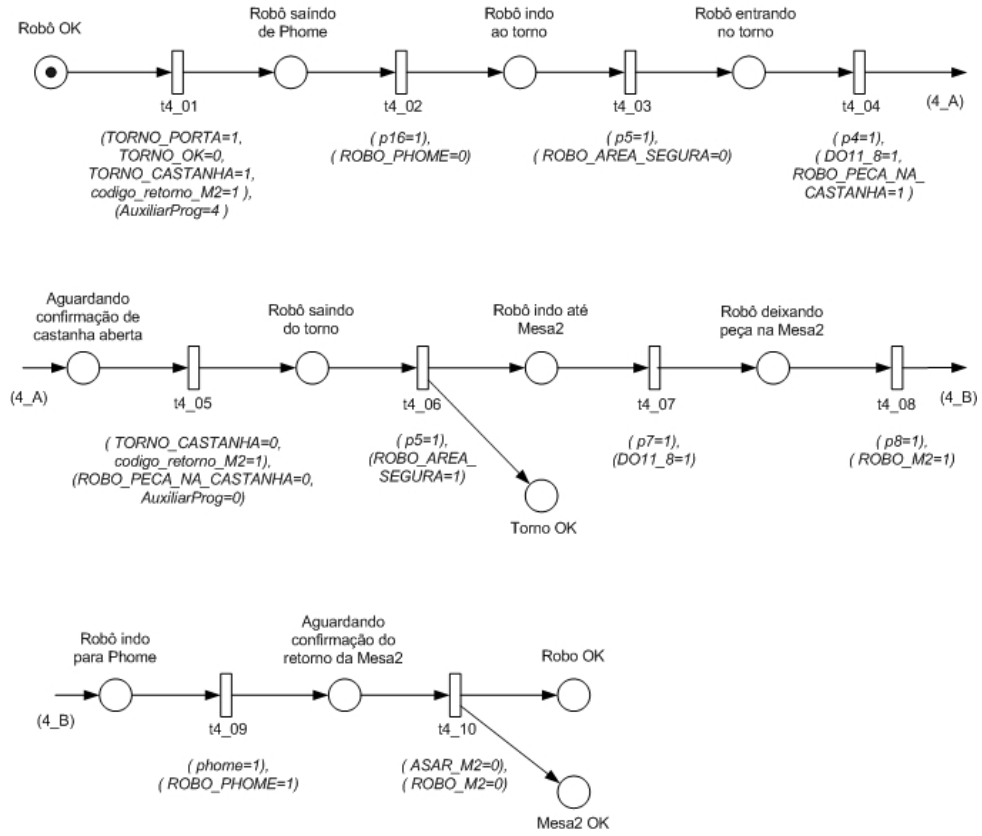




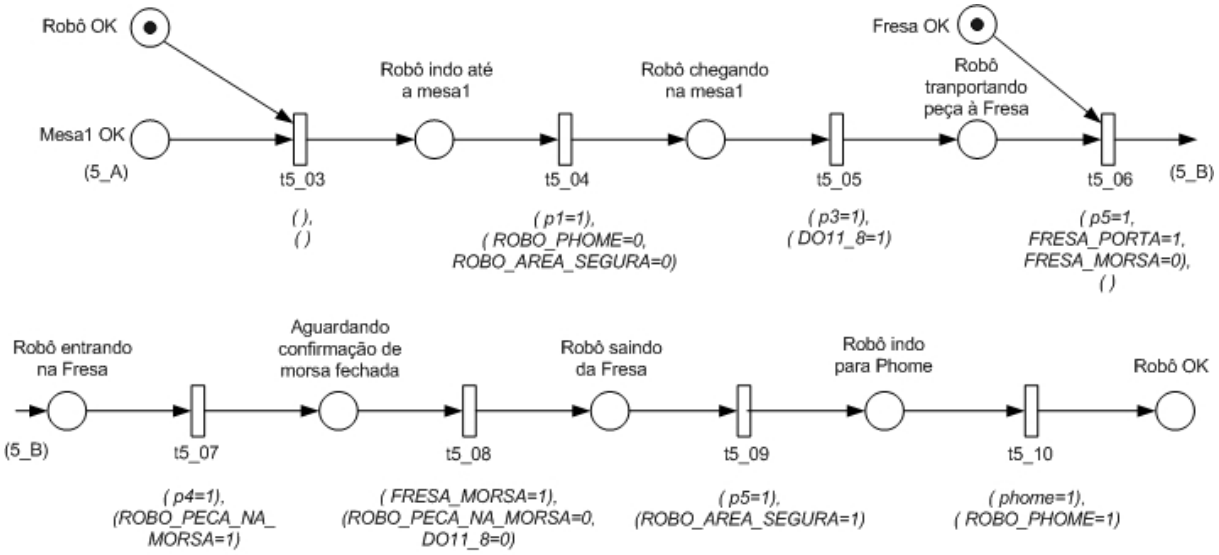
A8. Redes de Petri Interpretadas da integração do torno com a mesa1 do AS/RS

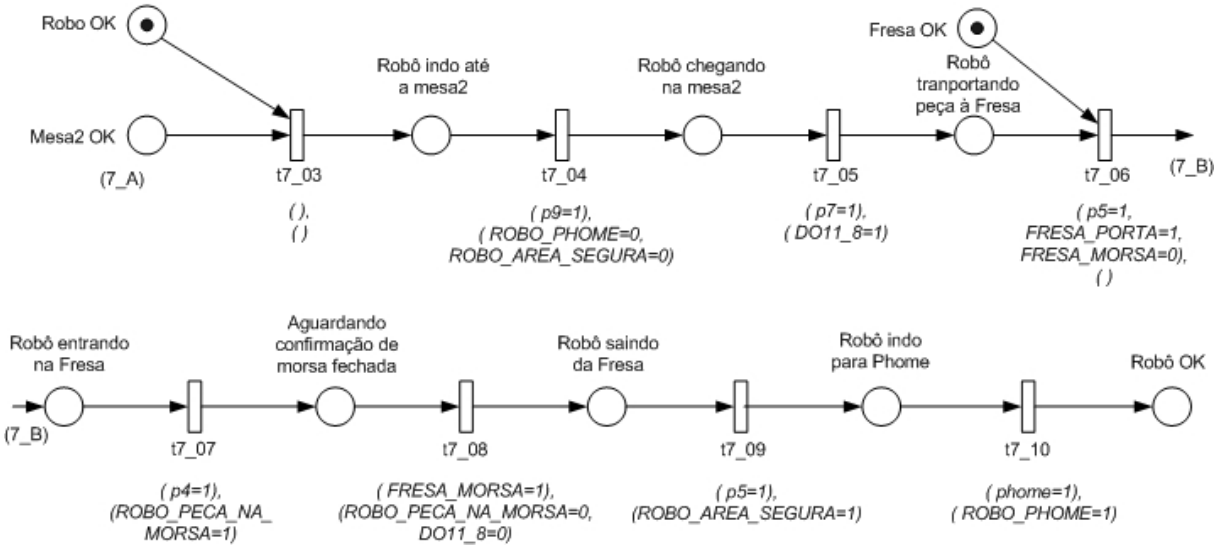


A9. Redes de Petri Interpretadas da integração do torno com a mesa2 do AS/RS

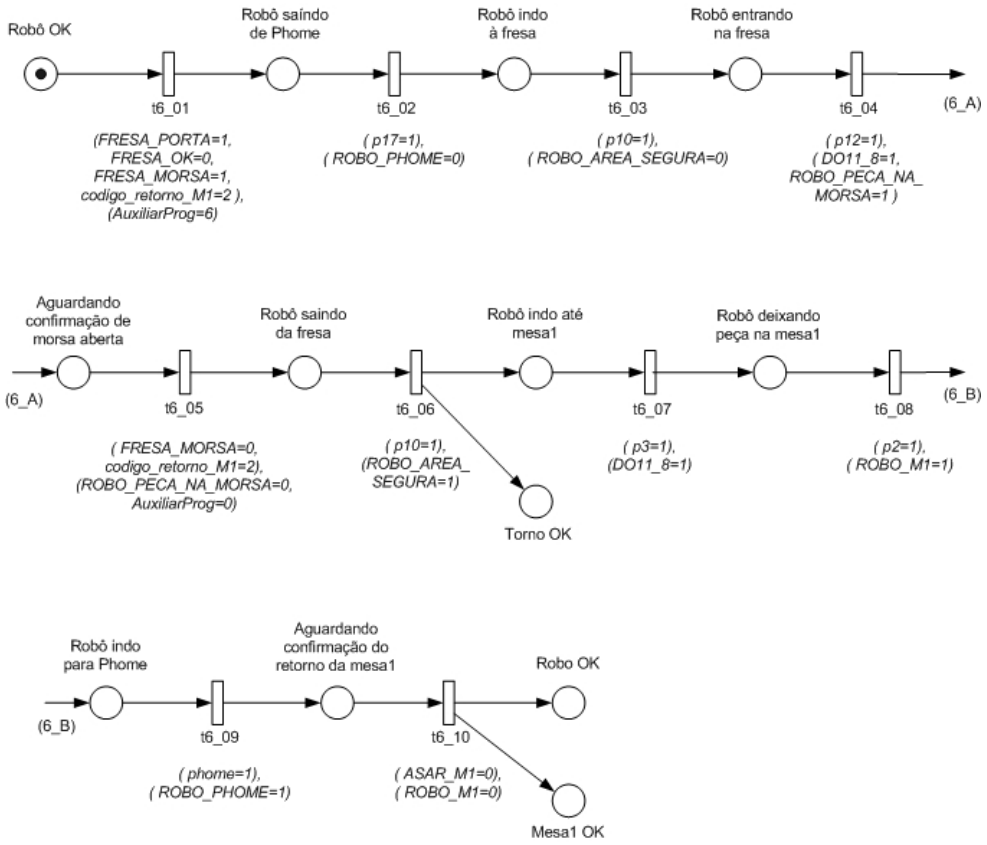


A10. Redes de Petri Interpretadas da integração das mesas do AS/RS com a fresadora

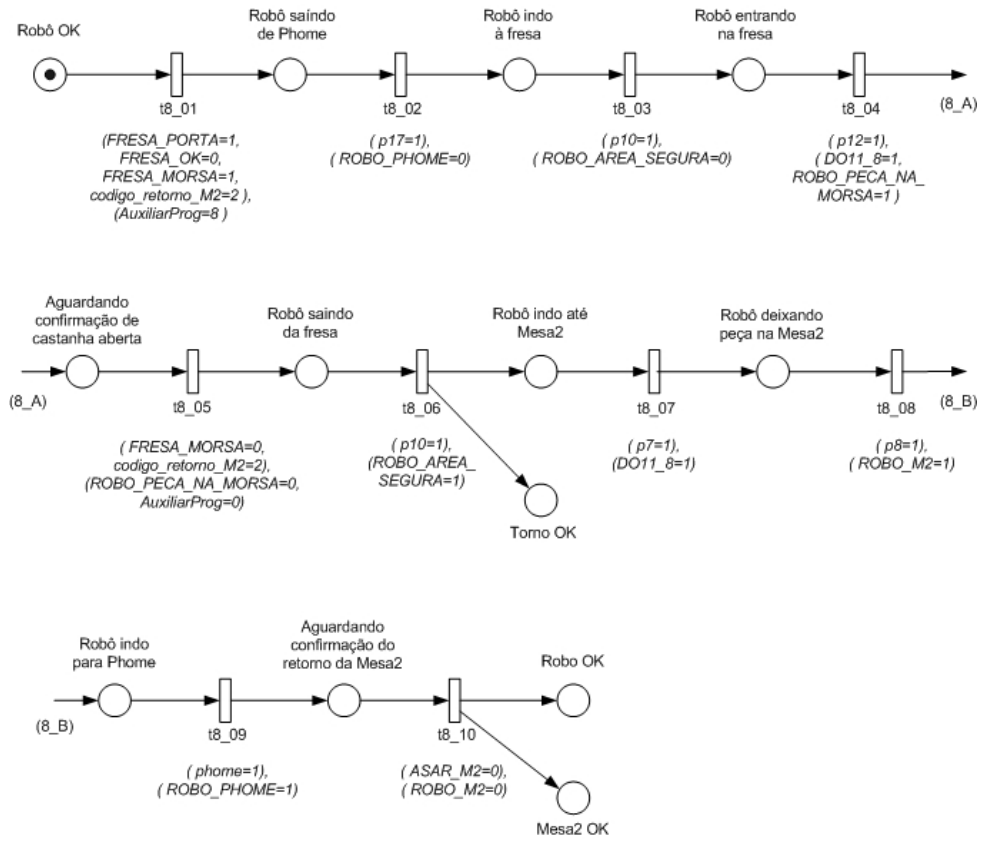




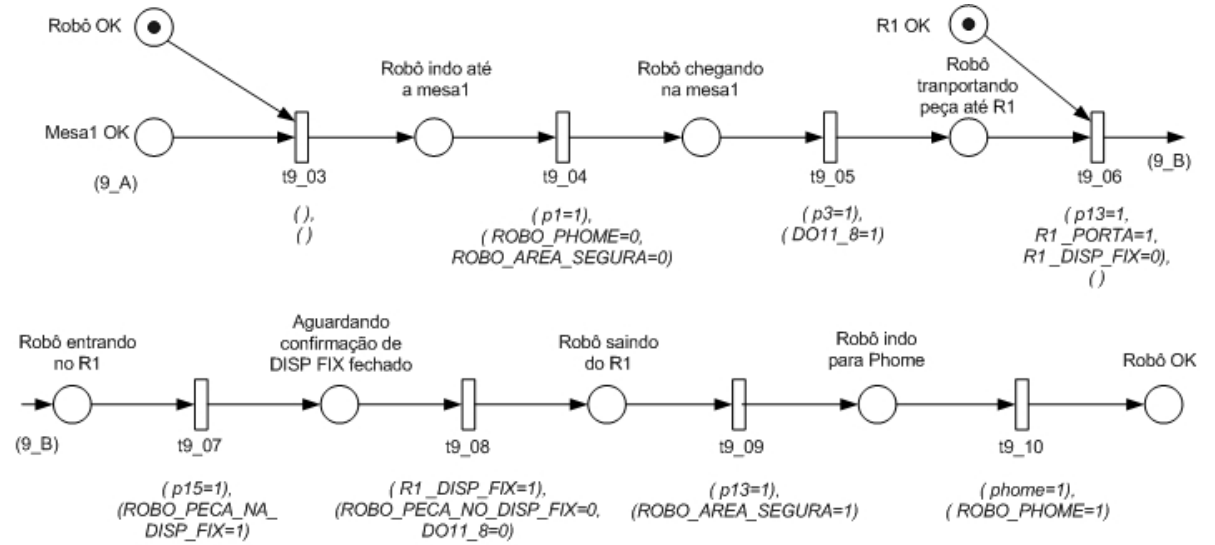
A11. Redes de Petri Interpretadas da integração da fresadora com a mesa1 do AS/RS

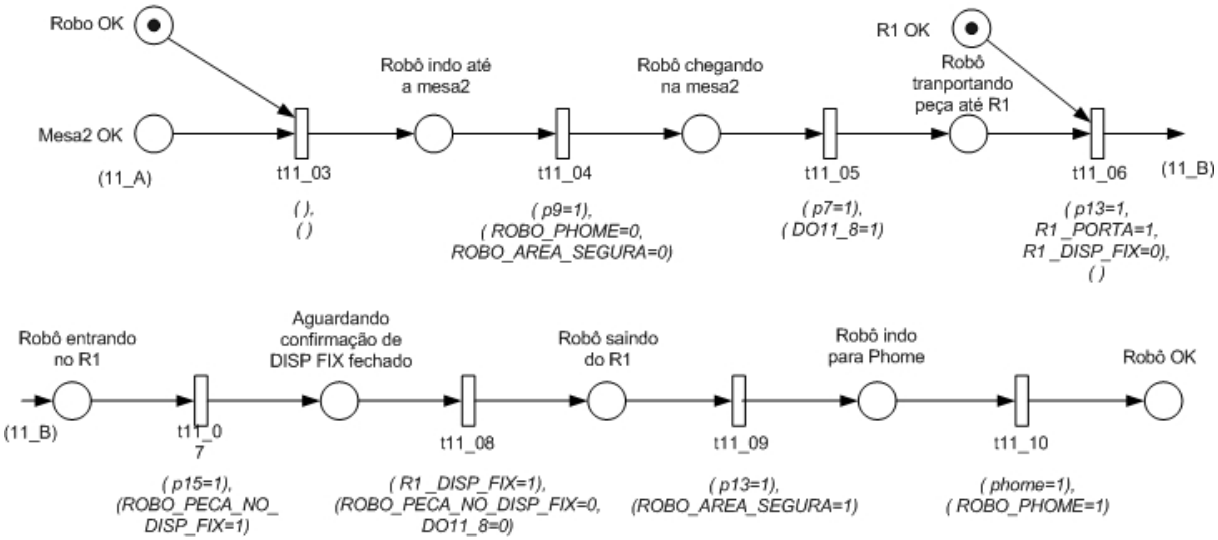


A12. Redes de Petri Interpretadas da integração da fresadora com a mesa2 do AS/RS

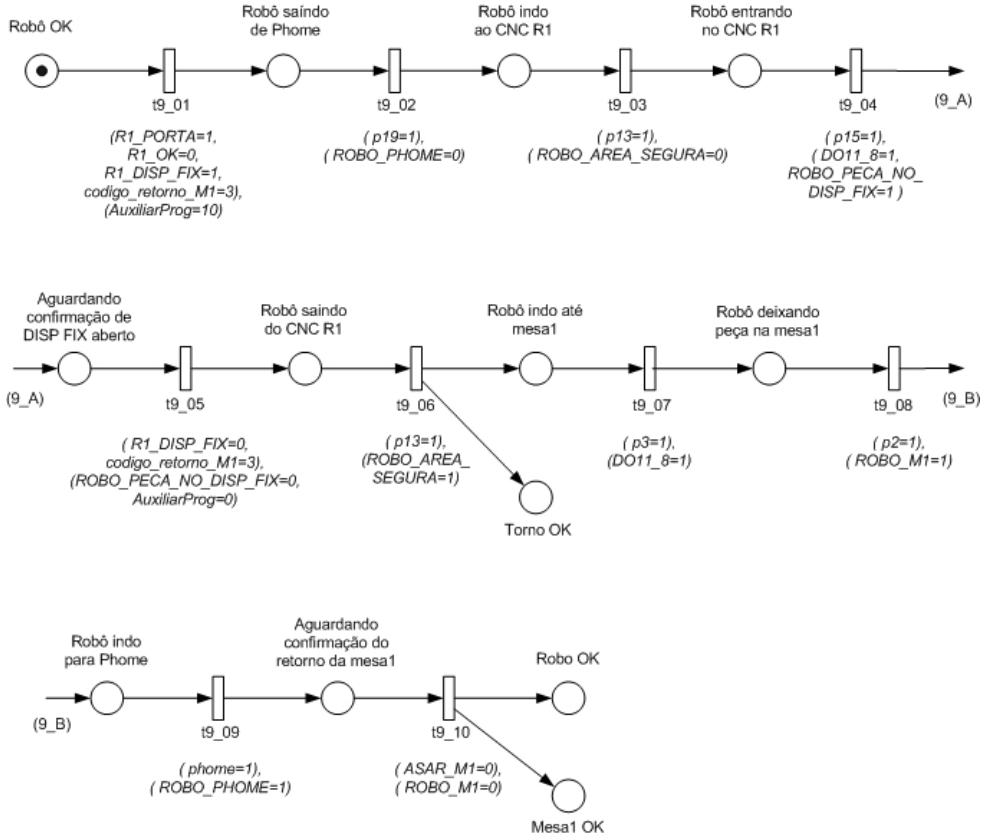


A13. Redes de Petri Interpretadas da integração das mesas do AS/RS com o CNC R1

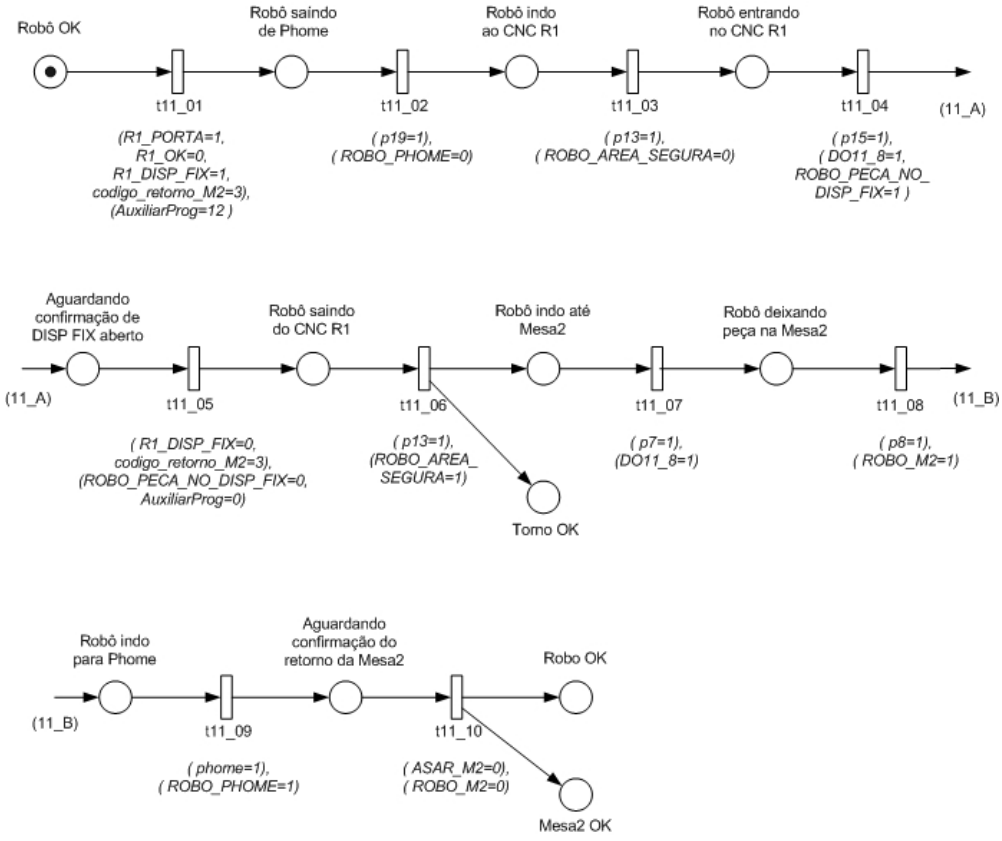




A14. Redes de Petri Interpretadas da integração do CNC R1 com a mesa 1 do AS/RS

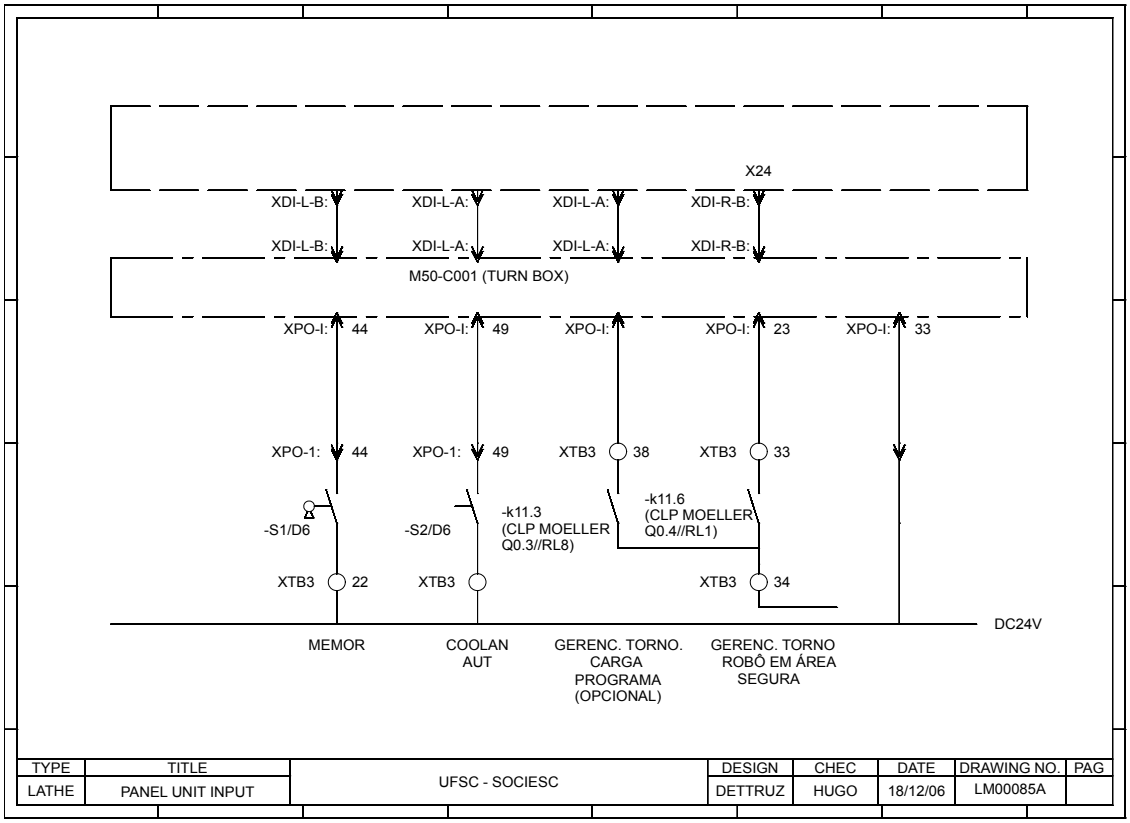


A15. Redes de Petri Interpretadas da integração do CNC R1 com a mesa 2 do AS/RS

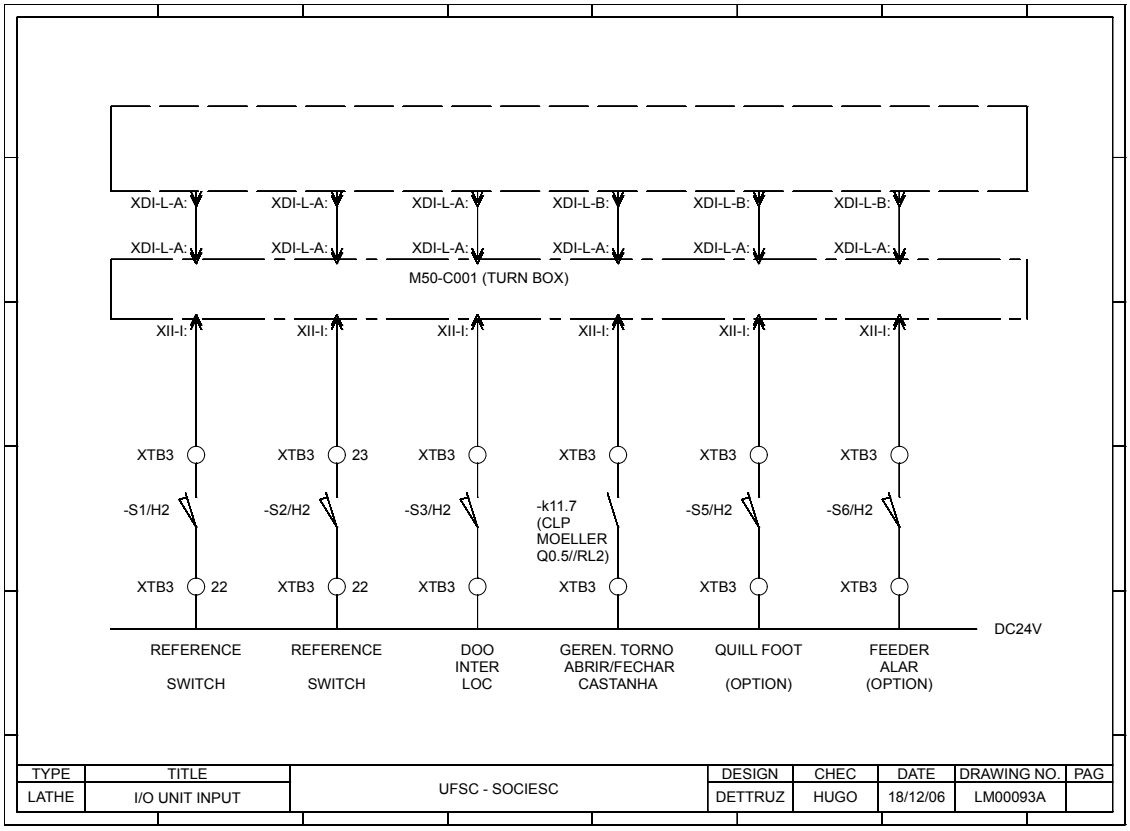


APÊNDICE B – DIAGRAMAS ELÉTRICOS MODIFICADOS NO TORNO

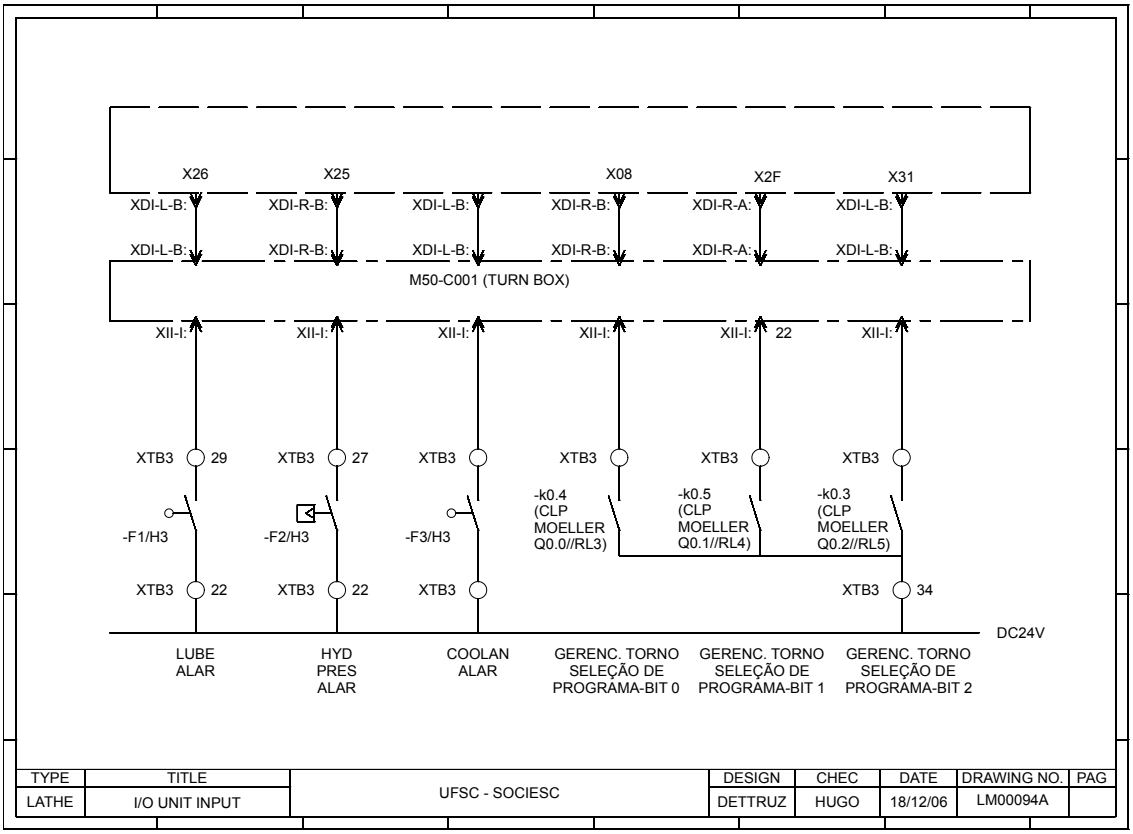
O Apêndice B apresenta os diagramas elétricos das modificações de *hardware* realizadas no torno CNC para possibilitar sua integração com o gerenciador.



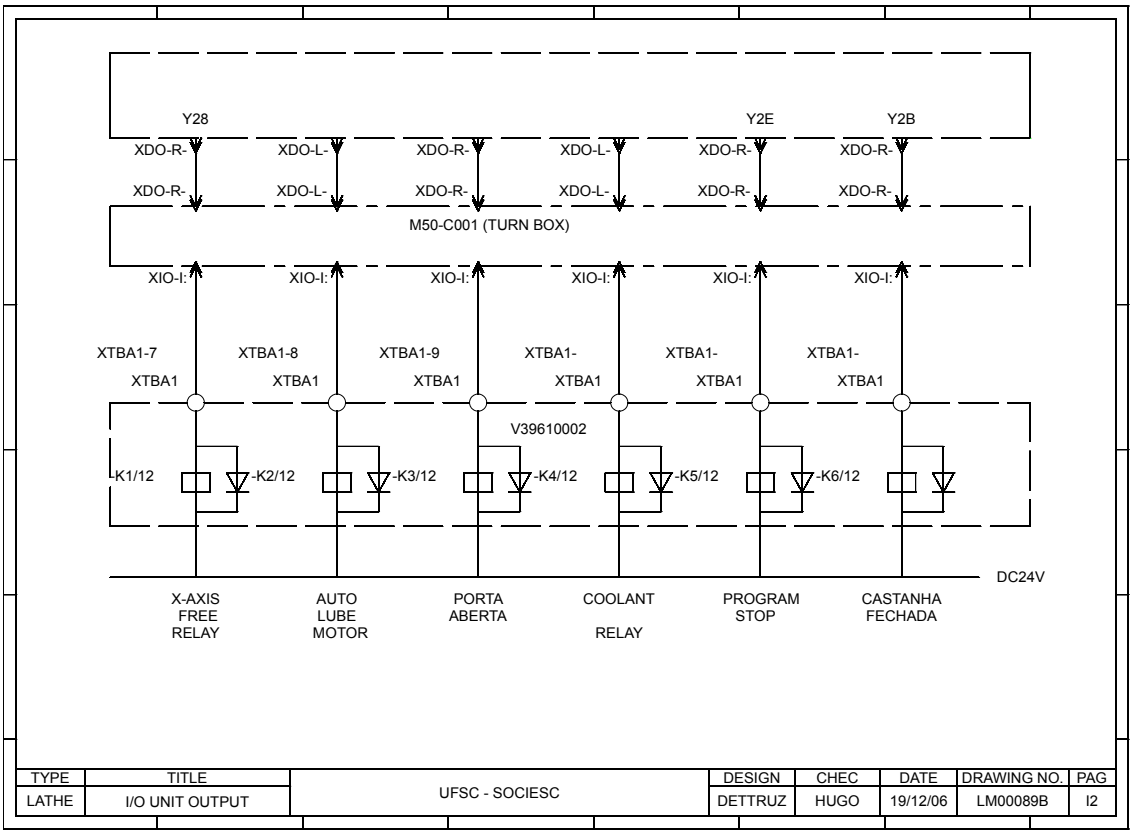
TYPE	TITLE		DESIGN	CHEC	DATE	DRAWING NO.	PAG
LATHE	PANEL UNIT INPUT	UFSC - SOCIESC	DETTRUZ	HUGO	18/12/06	LM00085A	



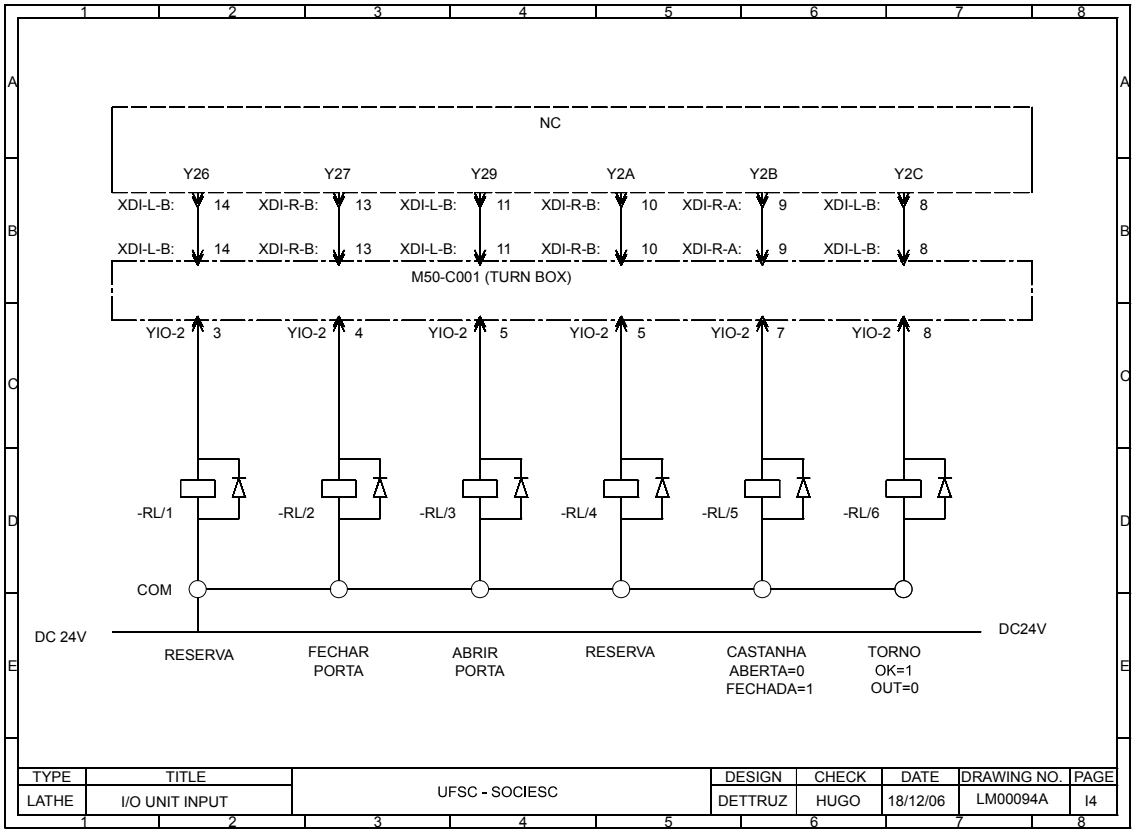
TYPE	TITLE		DESIGN	CHEC	DATE	DRAWING NO.	PAG
LATHE	I/O UNIT INPUT	UFSC - SOCIESC	DETTRUZ	HUGO	18/12/06	LM00093A	

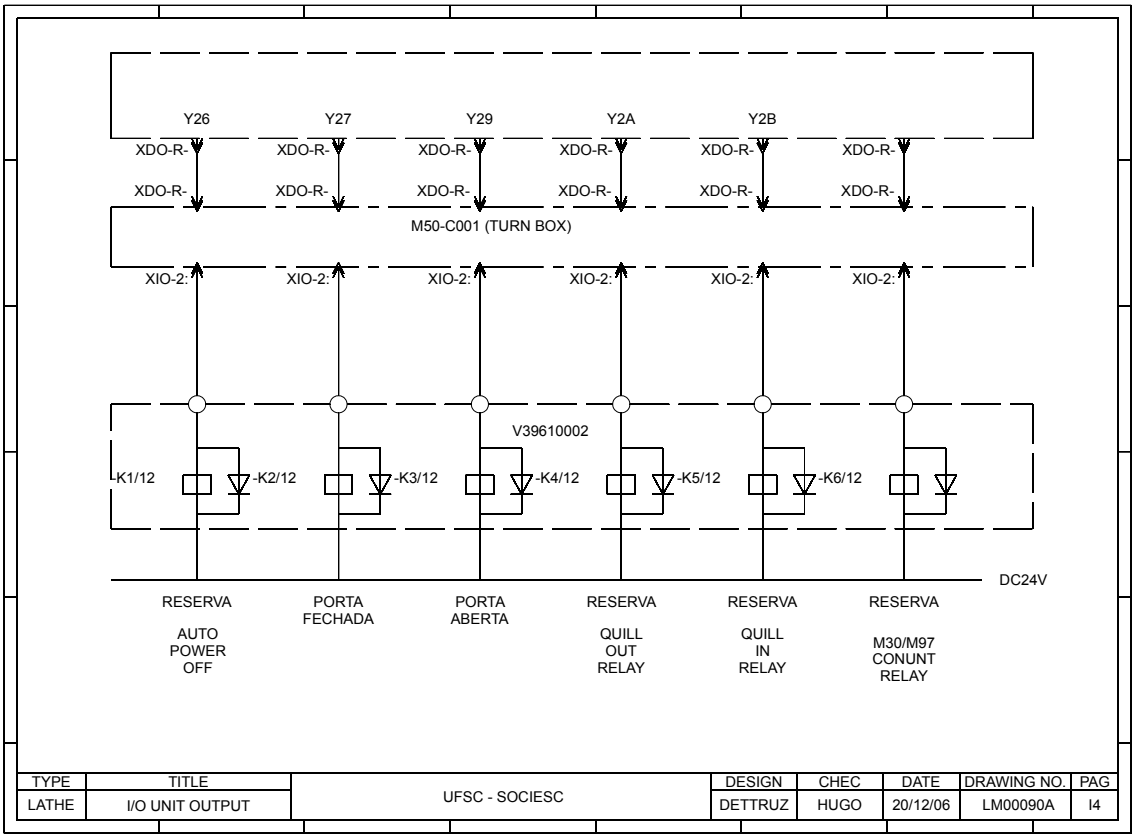


TYPE	TITLE		DESIGN	CHEC	DATE	DRAWING NO.	PAG
LATHE	I/O UNIT INPUT	UFSC - SOCIESC	DETTRUZ	HUGO	18/12/06	LM00094A	



TYPE	TITLE		DESIGN	CHEC	DATE	DRAWING NO.	PAG
LATHE	I/O UNIT OUTPUT	UFSC - SOCIESC	DETTRUZ	HUGO	19/12/06	LM00089B	I2


NC
Y26
Y27
Y29
Y2A
Y2B
Y2C
XDI-L-B: 14
XDI-R-B: 13
XDI-L-B: 11
XDI-R-B: 10
XDI-R-A: 9
XDI-L-B: 8
M50-C001 (TURN BOX)
YIO-2 3
YIO-2 4
YIO-2 5
YIO-2 5
YIO-2 7
YIO-2 8
-RL/1
-RL/2
-RL/3
-RL/4
-RL/5
-RL/6
COM
DC 24V
DC24V
RESERVA
FECHAR PORTA
ABRIR PORTA
RESERVA
CASTANHA ABERTA=0 FECHADA=1
TORNO OK=1 OUT=0
TYPE
TITLE
LATHE
I/O UNIT INPUT
UFSC - SOCIESC
DESIGN
CHECK
DATE
DRAWING NO.
PAGE
DETTRUZ
HUGO
18/12/06
LM00094A
I4


Y26
Y27
Y29
Y2A
Y2B
XDO-R-
M50-C001 (TURN BOX)
XIO-2:
V39610002
K1/12
-K2/12
-K3/12
-K4/12
-K5/12
-K6/12
DC24V
RESERVA
AUTO POWER OFF
PORTA FECHADA
PORTA ABERTA
RESERVA
QUILL OUT RELAY
RESERVA
QUILL IN RELAY
RESERVA
M30/M97 CONUNT RELAY
TYPE
TITLE
LATHE
I/O UNIT OUTPUT
UFSC - SOCIESC
DESIGN
CHEC
DATE
DRAWING NO.
PAG
DETTRUZ
HUGO
20/12/06
LM00090A
I4

APÊNDICE C – PROGRAMAÇÃO DO ROBÔ

O Apêndice C apresenta informações complementares sobre a movimentação do robô como: a seqüência de pontos na trajetória de cada programa, o fluxograma dos programas de movimentação e os próprios programas. Linhas de comentários, precedidos do símbolo !, foram inseridos ao longo de todo o programa para facilitar o entendimento.

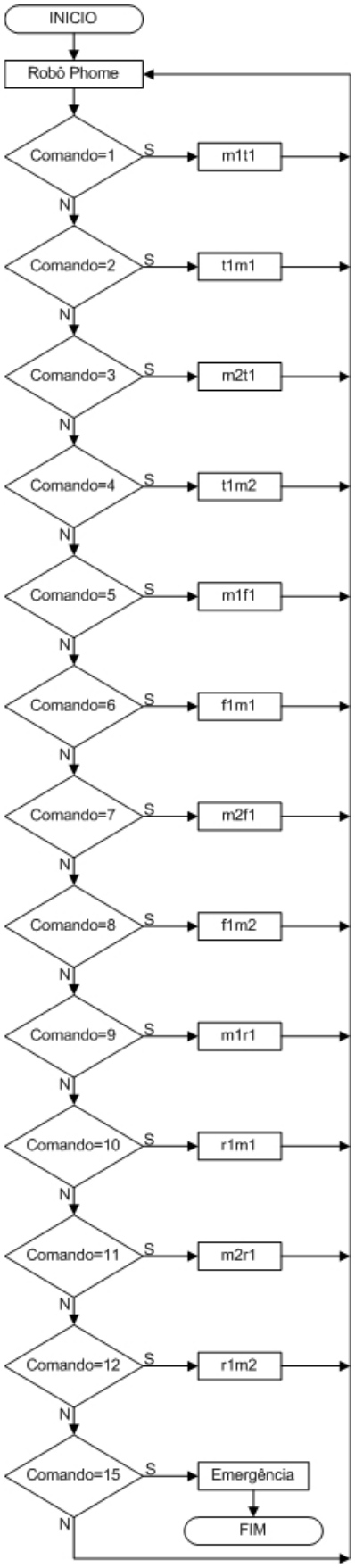
C1. Tabela da seqüência de pontos dos programas do robô

COMANDO	DECIMAL	PROGRAMA	SEQÜÊNCIA DE PONTOS
0000	0	Nenhum	
0001	1	m1t1	Phome-P1-P2-P3-P2-P16-P5-P6-P4-P6-P5-Phome
0010	2	t1m1	Phome-P5-P6-P4-P6-P5-P16-P2-P3-P2-P1-Phome
0011	3	m2t1	Phome-P9-P8-P7-P8-P2-P16-P5-P16-P4-P6-P5-Phome
0100	4	t1m2	Phome-P5-P6-P4-P6-P5-P16-P2-P8-P7-P8-P9-Phome
0101	5	m1f1	Phome-P1-P2-P3-P2-P16-P17-P18-P10-P11-P12-P11-P10-Phome
0110	6	f1m1	Phome-P10-P11-P12-P11-P10-P17-P16-P2-P3-P2-P1-Phome
0111	7	m2f1	Phome-P9-P8-P7-P8-P2-P16-P17-P18-P10-P11-P12-P11-P10-Phome
1000	8	f1m2	Phome-P10-P11-P12-P11-P10-P18-P17-P16-P2-P8-P7-P8-P9-Phome
1001	9	m1r1	Phome-P1-P2-P3-P2-P8-P19-P13-P14-P15-P14-P13-Phome
1010	10	r1m1	Phome-P13-P14-P15-P14-P13-P19-P8-P2-P3-P2-P1-Phome
1011	11	m2r1	Phome-P9-P8-P7-P18-P19-P13-P14-P15-P14-P13-Phome
1100	12	r1m2	Phome-P13-P14-P15-P14-P13-P19-P8-P7-P9-P9-Phome
1101	13	Reserva	Nenhum
1110	14	Reserva	Nenhum
1111	15	Emerg	Nenhum

C2. Posições relevantes da movimentação do robô

Posições relevantes da movimentação do robô	
P3	Robô pegando/devolvendo peça no *pallet* da Mesa1
P7	Robô pegando/devolvendo peça no *pallet* da Mesa2
P5	Robô em frente à porta do torno
P4	Robô colocando/retirando peça na castanha do torno
P10	Robô em frente à porta da fresadora
P12	Robô colocando/retirando peça na morsa da fresadora
P13	Robô em frente à porta do CNC R1
P15	Robô colocando/retirando peça no dispositivo de fixação de R1
P5, P10 e P13	Fronteiras da área de segurança do robô
Phome	Posição inicial do robô

C3. Fluxograma dos programas de movimentação do robô



C4. Programas do robô

```
%%%
  VERSION:1
  LANGUAGE:ENGLISH
%%%

MODULE HUGO
  CONST robtarget p18:=[[606.62,-462.36,1011.12],[0.010666,0.579894,0.814622,-0.00032],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget
p17:=[[762.01,32.87,1011.12],[0.010124,0.26815,0.963319,0.003318],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E
+09,9E+09]];
  CONST robtarget p19:=[[-804.01,833.68,957.32],[0.009193,0.892006,-0.45169,-
0.014751],[1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p16:=[[534.43,544.17,1011.12],[0.008169,-
0.103929,0.994528,0.006807],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p20:=[[946.5,724.06,1010.96],[0.008553,-
0.028575,0.999536,0.006257],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p15:=[[-1178.18,185.54,763.9],[0.013379,0.989011,-0.146823,-
0.011068],[1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p14:=[[-1178.14,185.5,957.23],[0.013381,0.989007,-0.146849,-
0.01109],[1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p13:=[[-973.97,185.5,957.29],[0.013372,0.989007,-0.146849,-
0.011089],[1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p12:=[[37.87,-1241.24,840.09],[0.009979,0.817031,0.576495,-0.003878],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p11:=[[37.91,-1241.2,1013.62],[0.009953,0.817045,0.576475,-0.003867],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p10:=[[33.09,-643.98,1013.41],[0.009941,0.819173,0.573448,-0.00391],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p9:=[[-152.62,1102.47,1287.64],[0.007688,-0.712373,0.701718,0.007661],[1,-
1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p8:=[[-152.67,1148.05,957.37],[0.007771,-0.712373,0.701716,0.007652],[1,-
1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p7:=[[-152.67,1148,827.39],[0.007788,-0.712377,0.701713,0.007638],[1,-
1,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p6:=[[1162.06,-23.74,650.85],[0.698281,0.160342,0.68203,0.146729],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p5:=[[845.68,-20.14,1097.61],[0.685335,0.210388,0.66851,0.197859],[-
1,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget
p4:=[[1167.28,46.67,650.74],[0.695076,0.145843,0.685331,0.161001],[0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E
+09,9E+09]];
  CONST robtarget p3:=[[210.43,1156.81,783.2],[0.007852,-0.707042,0.707087,0.007657],[0,0,-
1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p2:=[[210.41,1153.81,943.03],[0.007773,-0.707015,0.707116,0.007618],[0,0,-
1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST robtarget p1:=[[210.47,1098.33,1275.66],[0.007724,-0.707031,0.707099,0.00761],[0,0,-
1,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];
  CONST jointtarget jposhome:=[[90,-30,0,0,0,0],[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09]];

PROC m1t1()
    !Verifica se o torno está OK
   IF DI10_1=1 THEN
    ! inicio do programa mesa1-torno1
    MoveAbsJ jposhome,v200,z80,tool0;
    MoveJ p1,v200,z80,tool0;
    ! informa que robô está fora de Phome
    Reset DO10_1;
    MoveJ p2,v200,z80,tool0;
    ! informa que robô está fora de Área Segura
    Reset DO10_6;
    MoveL p3,v100,fine,tool0;
    WaitTime 1;
    ! fecha a garra para pegar a peça no palete da mesa1
    Set DO11_8;
    WaitTime 1;
    MoveJ p2,v200,z80,tool0;
    SingArea\Wrist;
```

```
  MoveJ p16,v200,fine,tool0;
  MoveL p5,v200,fine,tool0;
  ! espera sinal de confirmação da porta do torno aberta
  WaitDI DI10_3,1;
  ! espera sinal de confirmação da castanha aberta
  WaitDI DI10_2,0;
  MoveL p6,v200,fine,tool0;
  MoveL p4,v80,fine,tool0;
  ! envia sinal para fechar a castanha (pulso)
  Set DO10_2;
  ! espera sinal de castanha fechada
  WaitDI DI10_2,1;
  ! encerra pulso para fechar a castanha
  Reset DO10_2;
  ! abre a garra para deixar a peça no torno
  Reset DO11_8;
  WaitTime 1;
  MoveL p6,v200,fine,tool0;
  MoveL p5,v200,fine,tool0;
  ! envia sinal de robô em área segura
  Set DO10_6;
  MoveAbsJ jposhome,v200,z80,tool0;
  ! envia sinal de robô em Phome
  Set DO10_1;
 ENDIF
ENDPROC

PROC t1m1()
  !Verifica se o torno está OK
 IF DI10_1=1 THEN
  ! inicio do programa torno1-mesa1
  ! espera sinal de porta aberta
  WaitDI DI10_3,1;
  SingArea\Wrist;
  MoveJ p16,v200,z100,tool0;
  ! informa que robô está fora de Phome
  Reset DO10_1;
  MoveL p5,v200,fine,tool0;
  ! informa que robô está fora de Área Segura
  Reset DO10_6;
  MoveL p6,v200,fine,tool0;
  MoveL p4,v80,fine,tool0;
  WaitTime 1;
  ! fecha a garra para pegar peça
  Set DO11_8;
  WaitTime 1;
  ! envia pulso para abrir castanha
  Set DO10_2;
  ! espera sinal de castanha aberta
  WaitDI DI10_2,0;
  ! encerra pulso para fechar a castanha
  Reset DO10_2;
  MoveL p6,v200,fine,tool0;
  MoveL p5,v200,fine,tool0;
  ! envia sinal de robô em área segura
  Set DO10_6;
  MoveJ p16,v200,z100,tool0;
  MoveJ p2,v200,z100,tool0;
  MoveL p3,v100,fine,tool0;
  WaitTime 1;
  ! devolve a peça para o palete 1
  Reset DO11_8;
  WaitTime 1;
  MoveJ p2,v200,z80,tool0;
  ! envia sinal ao AS/AR para retorno do palete mesa1
  Set DO10_4;
  MoveJ p1,v200,z80,tool0;
  MoveAbsJ jposhome,v200,z80,tool0;
  ! envia sinal de robô em Phome
  Set DO10_1;
  ! aguarda confirmação do AS/AR de retorno da M1
  WaitDI DI10_7,0;
  ! encerra pulso para retorno da M1
  Reset DO10_4;
 ENDIF
ENDPROC

PROC m2t1()
  ! Verifica se o torno está OK
 IF DI10_1=1 THEN
  ! inicio do programa mesa2-torno1
  MoveAbsJ jposhome,v200,z80,tool0;
  SingArea\Wrist;
  MoveL p9,v200,z100,tool0;
  ! informa que robô está fora de Phome
  Reset DO10_1;
  MoveL p8,v200,z100,tool0;
  ! informa que robô está fora de Área Segura
  Reset DO10_6;
  MoveJ p7,v100,fine,tool0;
  WaitTime 1;
  ! fecha a garra para pegar a peça no palete 2
  Set DO11_8;
  WaitTime 1;
  MoveL p8,v200,fine,tool0;
  MoveJ p2,v200,z80,tool0;
  MoveJ p16,v200,fine,tool0;
  SingArea\Wrist;
  MoveL p5,v200,fine,tool0;
  ! espera sinal de confirmacao de porta aberta do torno
  WaitDI DI10_3,1;
  ! espera sinal de confirmação da castanha aberta
  WaitDI DI10_2,0;
  MoveL p6,v200,fine,tool0;
  MoveL p4,v80,fine,tool0;
  ! envia pulso para fechar a castanha
  Set DO10_2;
  ! espera confirmação de castanha fechada
  WaitDI DI10_2,1;
  ! encerra pulso para fechar a castanha
  Reset DO10_2;
  ! abre a garra para deixar a peça no torno
  Reset DO11_8;
  WaitTime 1;
  MoveL p6,v200,fine,tool0;
  MoveL p5,v200,fine,tool0;
  ! indica robô em área segura
  Set DO10_6;
  MoveAbsJ jposhome,v200,z80,tool0;
  ! envia sinal de robô em Phome
  Set DO10_1;
 ENDIF
ENDPROC

PROC t1m2()
  ! Verifica se o torno está OK
 IF DI10_1=1 THEN
  !  inicio do programa torno1-mesa2
```

```
  ! espera sinal de porta aberta
  WaitDI DI10_3,1;
  SingArea\Wrist;
  MoveJ p16,v200,z80,tool0;
  ! informa que robô está fora de Phome
  Reset DO10_1;
  MoveL p5,v200,z80,tool0;
  ! informa que robô está fora de Área Segura
  Reset DO10_6;
  MoveL p6,v200,fine,tool0;
  MoveL p4,v80,fine,tool0;
  WaitTime 1;
  ! fecha a garra para pegar a peça
  Set DO11_8;
  WaitTime 1;
  ! envia pulso para abrir a castanha
  Set DO10_2;
  ! espera sinal de castanha aberta
  WaitDI DI10_2,0;
  ! encerra pulso para fechar a castanha
  Reset DO10_2;
  WaitTime 1;
  MoveL p6,v200,fine,tool0;
  MoveL p5,v200,fine,tool0;
  ! envia sinal de robô em área segura
  Set DO10_6;
  MoveJ p16,v200,z100,tool0;
  MoveJ p2,v200,z100,tool0;
  MoveL p8,v200,z100,tool0;
  MoveJ p7,v100,fine,tool0;
  WaitTime 1;
  ! abre a garra para devolver a peça ao palete mesa 2
  Reset DO11_8;
  WaitTime 1;
  MoveL p8,v200,fine,tool0;
  ! envia sinal ao AS/AR para retorno do palete mesa 2
  Set DO10_5;
  MoveL p9,v200,z100,tool0;
  MoveAbsJ jposhome,v200,z80,tool0;
  ! envia sinal de robô em Phome
  Set DO10_1;
  ! aguarda confirmação do AS/AR de retorno da M2
  WaitDI DI10_8,0;
  ! encerra pulso para retorno da M1
  Reset DO10_5;
 ENDIF
ENDPROC

PROC m1f1()
  !Verifica se a fresadora está OK
 IF DI10_4=1 THEN
  ! inicio do programa mesa1-fresa1
  MoveAbsJ jposhome,v200,z80,tool0;
  MoveJ p1,v200,z100,tool0;
  ! informa que robô está fora de Phome
  Reset DO10_1;
  MoveJ p2,v200,z100,tool0;
  ! informa que robô está fora de Área Segura
  Reset DO10_6;
  MoveL p3,v100,fine,tool0;
  WaitTime 1;
  ! fecha a garra para pegar a peça no palete da mesa1
  Set DO11_8;
  WaitTime 1;
  MoveJ p2,v200,z80,tool0;
  SingArea\Wrist;
  MoveJ p16,v200,z80,tool0;
  MoveJ p17,v200,z80,tool0;
  MoveJ p18,v200,z80,tool0;
  MoveL p10,v200,fine,tool0;
  ! espera sinal de confirmação da porta da fresa aberta
  WaitDI DI10_6,1;
  ! espera sinal de confirmação da morsa da fresa aberta
  WaitDI DI10_5,0;
  MoveL p11,v200,fine,tool0;
  MoveL p12,v100,fine,tool0;
  ! envia sinal para fechar a morsa da fresadora (pulso)
  Set DO10_3;
  ! espera sinal de morsa fechada
  WaitDI DI10_5,1;
  ! encerra pulso para fechar a morsa
  Reset DO10_3;
  ! abre a garra para deixar a peça na fresadora
  Reset DO11_8;
  WaitTime 1;
  MoveL p11,v200,fine,tool0;
  MoveL p10,v200,fine,tool0;
  ! envia sinal de robô em área segura
  Set DO10_6;
  MoveAbsJ jposhome,v200,z80,tool0;
  ! envia sinal de robô em Phome
  Set DO10_1;
 ENDIF
ENDPROC

PROC f1m1()
  !Verifica se a fresadora está OK
 IF DI10_4=1 THEN
  ! inicio do programa fresa1-mesa1
  ! espera sinal de porta aberta
  WaitDI DI10_6,1;
  MoveJ p17,v200,z100,tool0;
  ! informa que robô está fora de Phome
  Reset DO10_1;
  MoveL p18,v200,z100,tool0;
  MoveL p10,v200,z80,tool0;
  ! informa que robô está fora de Área Segura
  Reset DO10_6;
  MoveL p11,v200,fine,tool0;
  MoveL p12,v100,fine,tool0;
  WaitTime 1;
  ! fecha a garra para pegar peça
  Set DO11_8;
  WaitTime 1;
  ! envia pulso para abrir a morsa da fresadora
  Set DO10_3;
  ! espera sinal de morsa aberta
  WaitDI DI10_5,0;
  ! encerra pulso para fechar a morsa
  Reset DO10_3;
  MoveL p11,v200,fine,tool0;
  MoveL p10,v200,fine,tool0;
  ! informa que robô está em Área Segura
  Set DO10_6;
  MoveL p18,v200,z80,tool0;
  MoveL p17,v200,z80,tool0;
  MoveL p16,v200,z80,tool0;
  MoveJ p2,v200,z80,tool0;
```

```
  MoveL p3,v100,fine,tool0;
  WaitTime 1;
  ! devolve a peça para o palete 1
  Reset DO11_8;
  WaitTime 1;
  MoveJ p2,v200,z100,tool0;
  ! envia sinal ao AS/AR para retorno do palete mesa1
  Set DO10_4;
  MoveJ p1,v200,z100,tool0;
  MoveAbsJ jposhome,v200,z80,tool0;
  ! envia sinal de robô em Phome
  Set DO10_1;
  ! aguarda confirmação do AS/AR de retorno da M1
  WaitDI DI10_7,0;
  ! encerra pulso para retorno da M1
  Reset DO10_4;
 ENDIF
ENDPROC

PROC m2f1()
  ! Verifica se a fresadora está OK
 IF DI10_4=1 THEN
  ! inicio do programa mesa2-fresa1
  MoveAbsJ jposhome,v200,z80,tool0;
  SingArea\Wrist;
  MoveL p9,v200,z100,tool0;
  ! informa que robô está fora de Phome
  Reset DO10_1;
  MoveL p8,v200,z80,tool0;
  ! informa que robô está fora de Área Segura
  Reset DO10_6;
  MoveJ p7,v100,fine,tool0;
  WaitTime 1;
  ! fecha a garra para pegar a peça no palete 2
  Set DO11_8;
  WaitTime 1;
  MoveL p8,v200,fine,tool0;
  MoveJ p2,v200,z80,tool0;
  SingArea\Wrist;
  MoveJ p16,v200,z80,tool0;
  MoveJ p17,v200,z80,tool0;
  MoveJ p18,v200,z80,tool0;
  MoveL p10,v200,fine,tool0;
  ! espera sinal de confirmação da porta da fresa aberta
  WaitDI DI10_6,1;
  ! espera sinal de confirmação da morsa da fresa aberta
  WaitDI DI10_5,0;
  MoveL p11,v200,fine,tool0;
  MoveL p12,v100,fine,tool0;
  ! envia sinal para fechar a morsa da fresadora (pulso)
  Set DO10_3;
  ! espera sinal de morsa fechada
  WaitDI DI10_5,1;
  ! encerra pulso para fechar a morsa
  Reset DO10_3;
  ! abre a garra para deixar a peça na fresadora
  Reset DO11_8;
  WaitTime 1;
  MoveL p11,v200,fine,tool0;
  MoveL p10,v200,z80,tool0;
  ! envia sinal de robô em área segura
  Set DO10_6;
  MoveAbsJ jposhome,v200,z80,tool0;
  ! envia sinal de robô em Phome
  Set DO10_1;
 ENDIF
ENDPROC

PROC f1m2()
  ! Verifica se a fresadora está OK
 IF DI10_4=1 THEN
  ! inicio do programa fresa1-mesa2
  ! espera sinal de porta aberta da fresadora
  WaitDI DI10_6,1;
  SingArea\Wrist;
  MoveJ p17,v200,z80,tool0;
  ! informa que robô está fora de Phome
  Reset DO10_1;
  MoveL p18,v200,z80,tool0;
  MoveL p10,v200,fine,tool0;
  ! informa que robô está fora de Área Segura
  Reset DO10_6;
  MoveL p11,v200,fine,tool0;
  MoveL p12,v100,fine,tool0;
  WaitTime 1;
  ! fecha a garra para pegar peça
  Set DO11_8;
  WaitTime 1;
  ! envia pulso para abrir a morsa da fresadora
  Set DO10_3;
  ! espera sinal de morsa aberta
  WaitDI DI10_5,0;
  ! encerra pulso para fechar a morsa
  Reset DO10_3;
  MoveL p11,v200,fine,tool0;
  MoveL p10,v200,fine,tool0;
  ! informa que robô está em Área Segura
  Set DO10_6;
  MoveL p18,v200,z80,tool0;
  MoveL p17,v200,z80,tool0;
  MoveL p16,v200,z80,tool0;
  MoveJ p2,v200,z80,tool0;
  MoveL p8,v200,z80,tool0;
  MoveJ p7,v100,fine,tool0;
  WaitTime 1;
  ! abre a garra para devolver a peça ao palete mesa 2
  Reset DO11_8;
  WaitTime 1;
  MoveL p8,v200,z80,tool0;
  ! envia sinal ao AS/AR para retorno do palete mesa 2
  Set DO10_5;
  MoveL p9,v200,z100,tool0;
  MoveAbsJ jposhome,v200,z80,tool0;
  ! envia sinal de robô em Phome
  Set DO10_1;
  ! aguarda confirmação do AS/AR de retorno da M2
  WaitDI DI10_8,0;
  ! encerra pulso para retorno da M1
  Reset DO10_5;
 ENDIF
ENDPROC

PROC m1r1()
  !Verifica se o CNC1 está OK
 IF DI10_13=1 THEN
  ! inicio do programa mesa1-r1
  MoveAbsJ jposhome,v200,z80,tool0;
  MoveJ p1,v200,z80,tool0;
```

```
  ! informa que robô está fora de Phome
  Reset DO10_1;
  MoveJ p2,v200,z80,tool0;
  ! informa que robô está fora de Área Segura
  Reset DO10_6;
  MoveL p3,v100,fine,tool0;
  WaitTime 1;
  ! fecha a garra para pegar a peça no palete da mesa1
  Set DO11_8;
  WaitTime 1;
  MoveJ p2,v200,z80,tool0;
  MoveL p8,v200,z80,tool0;
  MoveJ p19,v200,z80,tool0;
  SingArea\Wrist;
  MoveL p13,v200,fine,tool0;
  ! espera sinal de confirmação porta do CNC1 aberta
  WaitDI DI10_15,1;
  ! espera sinal de confirmação do disp de fix aberto
  WaitDI DI10_14,0;
  MoveL p14,v200,fine,tool0;
  MoveL p15,v100,fine,tool0;
  ! envia sinal para fechar o dispositivo de fix do CNC1
  Set DO10_7;
  ! espera sinal de dispositivo fixação fechado
  WaitDI DI10_14,1;
  ! encerra pulso para fechar o dispositivo de fixação
  Reset DO10_7;
  ! abre a garra para deixar a peça no CNC1
  Reset DO11_8;
  WaitTime 1;
  MoveL p14,v200,fine,tool0;
  MoveL p13,v200,fine,tool0;
  ! envia sinal de robô em área segura
  Set DO10_6;
  MoveAbsJ jposhome,v200,z80,tool0;
  ! envia sinal de robô em Phome
  Set DO10_1;
 ENDIF
ENDPROC

PROC r1m1()
  !Verifica se o CNC1 está OK
 IF DI10_13=1 THEN
  ! inicio do programa r1-mesa1
  ! espera sinal de porta aberta
  WaitDI DI10_15,1;
  ! informa que robô está fora de Phome
  Reset DO10_1;
  SingArea\Wrist;
  MoveL p13,v200,z100,tool0;
  ! informa que robô está fora de Área Segura
  Reset DO10_6;
  MoveL p14,v200,fine,tool0;
  MoveL p15,v100,fine,tool0;
  WaitTime 1;
  ! fecha a garra para pegar peça
  Set DO11_8;
  WaitTime 1;
  ! envia pulso para abrir dispositivo de fixação
  Set DO10_7;
  ! espera sinal de dispositivo de fixação aberto
  WaitDI DI10_14,0;
  ! encerra pulso para fechar o dispositivo de fixação
  Reset DO10_7;
  MoveL p14,v200,fine,tool0;
  MoveL p13,v200,fine,tool0;
  ! informa que robô está em Área Segura
  Set DO10_6;
  MoveJ p19,v200,z80,tool0;
  MoveL p8,v200,z80,tool0;
  MoveJ p2,v200,z80,tool0;
  MoveL p3,v100,fine,tool0;
  WaitTime 1;
  ! devolve a peça para o palete 1
  Reset DO11_8;
  WaitTime 1;
  MoveJ p2,v200,z80,tool0;
  ! envia sinal ao AS/AR para retorno do palete mesa1
  Set DO10_4;
  MoveJ p1,v200,z80,tool0;
  MoveAbsJ jposhome,v200,z80,tool0;
  ! envia sinal de robô em Phome
  Set DO10_1;
  ! aguarda confirmação do AS/AR de retorno da M1
  WaitDI DI10_7,0;
  ! encerra pulso para retorno da M1
  Reset DO10_4;
 ENDIF
ENDPROC

PROC m2r1()
  ! Verifica se o CNC1 está OK
 IF DI10_13=1 THEN
  ! inicio do programa mesa2-r1
  MoveAbsJ jposhome,v200,z80,tool0;
  SingArea\Wrist;
  MoveL p9,v200,z80,tool0;
  ! informa que robô está fora de Phome
  Reset DO10_1;
  MoveL p8,v200,z80,tool0;
  ! informa que robô está fora de Área Segura
  Reset DO10_6;
  MoveJ p7,v100,fine,tool0;
  WaitTime 1;
  ! fecha a garra para pegar a peça no palete 2
  Set DO11_8;
  WaitTime 1;
  MoveL p8,v200,z80,tool0;
  MoveJ p9,v200,z80,tool0;
  SingArea\Wrist;
  MoveL p13,v200,fine,tool0;
  ! espera sinal de confirmação  porta do R11 aberta
  WaitDI DI10_15,1;
  ! espera sinal de confirmação do disp de fix aberto
  WaitDI DI10_14,0;
  MoveL p14,v200,fine,tool0;
  MoveL p15,v100,fine,tool0;
  ! envia sinal para fechar o disp de fixação do R1
  Set DO10_7;
  ! espera sinal de dispositivo fixação fechado
  WaitDI DI10_14,1;
  ! encerra pulso para fechar o dispositivo de fixação
  Reset DO10_7;
  ! abre a garra para deixar a peça no R1
  Reset DO11_8;
  WaitTime 1;
  MoveL p14,v200,fine,tool0;
  MoveL p13,v200,fine,tool0;
```

```
    ! envia sinal de robô em área segura
    Set DO10_6;
    MoveAbsJ jposhome,v200,z80,tool0;
    ! envia sinal de robô em Phome
    Set DO10_1;
  ENDIF
ENDPROC

PROC r1m2()
    ! Verifica se o CNC1 está OK
  IF DI10_13=1 THEN
    !  inicio do programa r1-mesa2
    ! espera sinal de porta aberta do R11
    WaitDI DI10_15,1;
    SingArea\Wrist;
    ! informa que robô está fora de Phome
    Reset DO10_1;
    MoveL p13,v200,fine,tool0;
    ! informa que robô está fora de Área Segura
    Reset DO10_6;
    MoveL p14,v200,fine,tool0;
    MoveL p15,v100,fine,tool0;
    WaitTime 1;
    ! fecha a garra para pegar a peça
    Set DO11_8;
    WaitTime 1;
    ! envia pulso para abrir o disp de fixação de R1
    Set DO10_7;
    ! espera sinal de dispositivo de fixação aberto
    WaitDI DI10_14,0;
    ! encerra pulso para fechar o dispositivo de fixação
    Reset DO10_7;
    WaitTime 1;
    MoveL p14,v200,fine,tool0;
    MoveL p13,v200,fine,tool0;
    ! informa que robô está em Área Segura
    Set DO10_6;
    MoveJ p19,v200,z80,tool0;
    MoveL p8,v200,z80,tool0;
    MoveJ p7,v100,fine,tool0;
    WaitTime 1;
    ! abre a garra para devolver a peça ao palete mesa 2
    Reset DO11_8;
    WaitTime 1;
    MoveL p8,v200,fine,tool0;
    ! envia sinal ao AS/AR para retorno do palete mesa 2
    Set DO10_5;
    MoveL p9,v200,z80,tool0;
    MoveAbsJ jposhome,v200,z80,tool0;
    ! envia sinal de robô em Phome
    Set DO10_1;
    ! aguarda confirmação do AS/AR de retorno da M2
    WaitDI DI10_8,0;
    ! encerra pulso para retorno da M1
    Reset DO10_5;
  ENDIF
ENDPROC

PROC emer()
  ! envia sinal de emergência
  Stop;
ENDPROC
```

```
  PROC main()
inicio:
    TEST Comando
    CASE 1:
      m1t1;
    CASE 2:
      t1m1;
    CASE 3:
      m2t1;
    CASE 4:
      t1m2;
    CASE 5:
      m1f1;
    CASE 6:
      f1m1;
    CASE 7:
      m2f1;
    CASE 8:
      f1m2;
    CASE 9:
      m1r1;
    CASE 10:
      r1m1;
    CASE 11:
      m2r1;
    CASE 12:
      r1m2;
    CASE 15:
      emer;
    CASE 0,13,14:
      GOTO inicio;
    ENDTEST
    GOTO inicio;
  ENDPROC
ENDMODULE
```

APÊNDICE D – *SCRIPTS* DE PROGRAMAÇÃO

O Apêndice D apresenta os principais *scripts* de programação dos gerenciadores do robô, da fresadora, do armazém, da FMC e o gerenciador remoto. Dos *scripts* de programação dos equipamentos CNC, apenas o da fresadora é apresentado. A construção dos gerenciadores do torno e do CNC R1 é semelhante ao da fresadora, respeitando as diferenças das *tags* definidas no capítulo 5 (tabelas 5.1 e 5.9).

D1. *SCRIPT* DE PROGRAMAÇÃO DO ROBÔ

```
//Gerenciador_Robo.Tags.AuxiliarProg (OnValueChanged)
//
SINAIS_PARA_ROBO2=AuxiliarProg+(ComunicR1_Robo*16)
//-----------------------------------------------------------------------------------------------------------

//Gerenciador_Robo.Tags.ENTRADAS.SINAIS_DO_ROBO.ROBO_PECA_NA_CASTANHA
(OnValueChanged)
 //Este script reseta o retorno de usinagem do torno
IF (codigo_retorno_M1 == 1 OR codigo_retorno_M2 == 1)
  AuxiliarProg=0
ENDIF
//-----------------------------------------------------------------------------------------------------------

//Gerenciador_Robo.Tags.ENTRADAS.SINAIS_DO_ROBO.ROBO_PECA_NA_MORSA
(OnValueChanged)
//Este script reseta o retorno de usinagem da fresa
IF (codigo_retorno_M1 == 2 OR codigo_retorno_M2 == 2)
  AuxiliarProg=0
ENDIF
//-----------------------------------------------------------------------------------------------------------

//Gerenciador_Robo.Tags.ENTRADAS.SINAIS_DO_ROBO.ROBO_PECA_NA_R1
(OnValueChanged)
 //Este script reseta o retorno de usinagem do CNC1
IF (codigo_retorno_M1 == 3 OR codigo_retorno_M2 == 3)
  AuxiliarProg=0
ENDIF
//-----------------------------------------------------------------------------------------------------------

//Gerenciador_Robô.OPCServer.ServerCNC1.ComunicR1_Robô (OnValueChanged)
//
SINAIS_PARA_ROBO2=AuxiliarProg+(ComunicR1_Robo*16)
//Este script seta o retorno da usinagem (prog r1m1 e r1m2) do CNC1 para M1 ou M2
IF (R1_OK == 0 AND R1_Disp_Fix == 1 AND R1_Porta == 1)
   IF codigo_retorno_M1 == 1
     AuxiliarProg=2    //seta o prog t1m1
   ELSEIF codigo_retorno_M2 == 1
     AuxiliarProg=4    //seta o prog t1m2
   ENDIF
ENDIF
```

```
//Gerenciador_Robô.OPCServer.ServerFresadora.ComunicFresa_Robô (OnValueChanged)
//
SINAIS_PARA_ROBO1=ComunicTorno_Robo+(ComunicFresa_Robo*8)+(presenca_mesa1*64)+(p
resenca_mesa2*128)
//Este script seta o retorno da usinagem (prog f1m1 e f1m2) da fresa para M1 ou M2
IF (FRESA_OK == 0 AND FRESA_MORSA == 1 AND FRESA_PORTA == 1)
   IF codigo_retorno_M1 == 2
     AuxiliarProg=6   //seta o prog m1f1
   ELSEIF codigo_retorno_M2 == 2
     AuxiliarProg=8  //seta o prog m2f1
   ENDIF
ENDIF
//-----------------------------------------------------------------------------------------------------------

//Gerenciador_Robô.OPCServer.ServerTorno.ComunicTorno_Robô (OnValueChanged)
//
SINAIS_PARA_ROBO1=ComunicTorno_Robo+(ComunicFresa_Robo*8)+(presenca_mesa1*64)+(p
resenca_mesa2*128)
//Este script seta o retorno da usinagem (prog t1m1 e t1m2) do torno para M1 ou M2
IF (TORNO_OK == 0 AND TORNO_CASTANHA == 1 AND TORNO_PORTA == 1)
   IF codigo_retorno_M1 == 1
     AuxiliarProg=2    //seta o prog t1m1
   ELSEIF codigo_retorno_M2 == 1
     AuxiliarProg=4    //seta o prog t1m2
   ENDIF
ENDIF
//-----------------------------------------------------------------------------------------------------------

//Gerenciador_Robô.OPCServer.ServerFMC.MovPeca (OnValueChanged)
//
IF MovPeca == "MP001"
  AuxiliarUsi=16
  AuxiliarProg=16
ENDIF
//-----------------------------------------------------------------------------------------------------------

//Gerenciador_Robô.OPCServer.ServerFMC.ProgUsinagem (OnValueChanged)
//
IF ProgUsinagem==" "
   AuxiliarUsi=0
   AuxiliarProg=0
ELSEIF ProgUsinagem=="PBR"
   AuxiliarUsi=1
   AuxiliarProg=1
ELSEIF ProgUsinagem=="TBR"
   AuxiliarUsi=2
   AuxiliarProg=7
ELSEIF ProgUsinagem=="CBR"
   AuxiliarUsi=3
   AuxiliarProg=5
ELSEIF ProgUsinagem=="BBR"
   AuxiliarUsi=4
   AuxiliarProg=3
ELSEIF ProgUsinagem=="DBR"
   AuxiliarUsi=5
   AuxiliarProg=1
ELSEIF ProgUsinagem=="RBR"
```

```
   AuxiliarUsi=6
   AuxiliarProg=7
ELSEIF ProgUsinagem=="PPR"
   AuxiliarUsi=7
   AuxiliarProg=1
ELSEIF ProgUsinagem=="TPR"
   AuxiliarUsi=8
   AuxiliarProg=7
ELSEIF ProgUsinagem=="CPR"
   AuxiliarUsi=9
   AuxiliarProg=5
ELSEIF ProgUsinagem=="BPR"
   AuxiliarUsi=10
   AuxiliarProg=3
ELSEIF ProgUsinagem=="DPR"
   AuxiliarUsi=11
   AuxiliarProg=1
ELSEIF ProgUsinagem=="RPR"
   AuxiliarUsi=12
   AuxiliarProg=7
ELSEIF ProgUsinagem=="M1R1"
   AuxiliarUsi=13
   AuxiliarProg=9
ELSEIF ProgUsinagem=="M2R1"
   AuxiliarUsi=14
   AuxiliarProg=11
ELSEIF ProgUsinagem=="EMERG"
   AuxiliarUsi=15
   AuxiliarProg=15
ELSEIF AuxiliarUsi=0
   AuxiliarProg=0
ENDIF
//-------------------------------------------------------------------------------------------------------------
```

//**Gerenciador_Robô.OPCServer.ServerFMC.s_scanner2** (OnValueChanged)

```
//
AuxiliarMesasUsinagem=s_scanner2
SINAIS_PARA_ROBO1=ComunicTorno_Robo+(ComunicFresa_Robo*8)+(presenca_mesa1*64)+(p
resenca_mesa2*128)
//-------------------------------------------------------------------------------------------------------------
```

D2. *SCRIPT* DE PROGRAMAÇÃO DA FRESADORA

// **Gerenciador_Fresadora.Tags.Fresa_Moeller** (OnValueChanged)

```
//
IF Fresa_Moeller<256
  FRESA_OK=0
  FRESA_CASTANHA=0
  FRESA_PORTA=0
  ComunicFresa_Robo=0
ENDIF
IF Fresa_Moeller>=256
  IF Fresa_Moeller<512
    FRESA_OK=1
    FRESA_CASTANHA=0
    FRESA_PORTA=0
    ComunicFresa_Robo=1
```

```
  ENDIF
ENDIF
IF Fresa_Moeller>=512
  IF Fresa_Moeller<768
    FRESA_OK=0
    FRESA_CASTANHA=1
    FRESA_PORTA=0
    ComunicFresa_Robo=2
  ENDIF
ENDIF
IF Fresa_Moeller>=768
  IF Fresa_Moeller<1024
    FRESA_OK=1
    FRESA_CASTANHA=1
    FRESA_PORTA=0
    ComunicFresa_Robo=3
  ENDIF
ENDIF
IF Fresa_Moeller>=1024
  IF Fresa_Moeller<1280
    FRESA_OK=0
    FRESA_CASTANHA=0
    FRESA_PORTA=1
    ComunicFresa_Robo=4
  ENDIF
ENDIF
IF Fresa_Moeller>=1280
  IF Fresa_Moeller<1536
    FRESA_OK=1
    FRESA_CASTANHA=0
    FRESA_PORTA=1
    ComunicFresa_Robo=5
  ENDIF
ENDIF
IF Fresa_Moeller>=1536
  IF Fresa_Moeller<1792
    FRESA_OK=0
    FRESA_CASTANHA=1
    FRESA_PORTA=1
    ComunicFresa_Robo=6
  ENDIF
ENDIF
IF Fresa_Moeller>=1792
  IF Fresa_Moeller<2048
    FRESA_OK=1
    FRESA_CASTANHA=1
    FRESA_PORTA=1
    ComunicFresa_Robo=7
  ENDIF
ENDIF
//-----------------------------------------------------------------------------------------------------------

// Gerenciador_Fresadora.OPCServers.ServerFMC.ProgUsinagem (OnValueChanged)
//
IF ProgUsinagem == " "
  AuxiliarProgFresa=0
  Bit0=0
  Bit1=0
  Bit2=0
ELSEIF ProgUsinagem == "TBR"
```

```
  AuxiliarProgFresa=1
  Bit0=1
  Bit1=1
  Bit2=1
ELSEIF ProgUsinagem == "TPR"
  AuxiliarProgFresa=2
  Bit0=1
  Bit1=1
  Bit2=1
ELSEIF ProgUsinagem == "CBR"
  AuxiliarProgFresa=3
  Bit0=1
  Bit1=0
  Bit2=1
ELSEIF ProgUsinagem == "CPR"
  AuxiliarProgFresa=4
  Bit0=1
  Bit1=0
  Bit2=1
ELSEIF ProgUsinagem == "RBR"
  AuxiliarProgFresa=5
  Bit0=1
  Bit1=1
  Bit2=1
ELSEIF ProgUsinagem == "RPR"
  AuxiliarProgFresa=6
  Bit0=1
  Bit1=1
  Bit2=1
ELSEIF ProgUsinagem=="EMERG"
  AuxiliarProgFresa=15
  Bit0=0
  Bit1=0
  Bit2=0
ELSEIF AuxiliarProgFresa=16
ENDIF
//-------------------------------------------------------------------------------------------------------------
```

//**Gerenciador_Fresadora.OPCServers.ServerRobo.SINAIS_DO_ROBO.ROBO_PECA_NA_MORSA** (OnValueChanged)

```
//
IF ROBO_PECA_NA_MORSA==0
  Morsa=0
ELSEIF ROBO_PECA_NA_MORSA==1
  Morsa=1
ENDIF
//-------------------------------------------------------------------------------------------------------------
```

//**Gerenciador_Fresadora.OPCServers.ServerRobo.SINAIS_DO_ROBO.ROBO_AREA_SEGURA** (OnValueChanged)

```
//
IF ROBO_AREA_SEGURA==0
  Area_Segura=0
ELSEIF ROBO_AREA_SEGURA==1
  Area_Segura=1
ENDIF
//-------------------------------------------------------------------------------------------------------------
```

D3. *SCRIPT* DE PROGRAMAÇÃO DO GERENCIADOR REMOTO

```
//Gerenciador_Remoto.Aplicação (OnStartRunning)
//
DIM tentativas
Login = 0
tentativas = 0
// Enquanto não se logar e não exceder três tentativas
WHILE Login == 0 AND tentativas < 3
  Login = Aplicação.Login()
  IF not Login // Se houver erro de login
    MessageBox("Login inválido!","ERRO")
  ENDIF
  tentativas +=1
WEND
IF Login ==  0 // Se excedeu as três tentativas
  Aplicação.StopRunning()   // Encerra a aplicação
ELSE // Se obteve sucesso no login
  RemotoUser=Aplicação.userLogin
  StatusRemoto1=1
  Tela1.Activate()
ENDIF
//-------------------------------------------------------------------------------------------------------------

//Gerenciador_Remoto.Aplicação (OnKeyPressEsc)
//
StatusRemoto1=0
RemotoUser=Aplicação.userLogin
Aplicação.StopRunning()
//-------------------------------------------------------------------------------------------------------------

//Gerenciador_Remoto.Tags.RemotoUser (OnValueChanged)
//
Aplicação.userLogin
//-------------------------------------------------------------------------------------------------------------

//Gerenciador_Remoto.Tags.TimeRemoto1 (OnIncrement)
//
AuxiliarCrono=180-TimeRemoto1.acum
//-------------------------------------------------------------------------------------------------------------

//Gerenciador_Remoto.Tags.TimeRemoto1 (OnPreset)
//
StatusRemoto1=0
Aplicação.StopRunning()
//-------------------------------------------------------------------------------------------------------------

//Gerenciador_Remoto.OPCServer.ServerFMC.MovEntrada (OnValueChanged)
//Gerenciador_Remoto.OPCServer.ServerFMC.MovSaida (OnValueChanged)
//Gerenciador_Remoto.OPCServer.ServerFMC.MovUsinagem (OnValueChanged)
//
IF MovUsinagem == 1
  StatusPulgus=1
  Pedido.visible=0
ENDIF
```

```
IF MovEntrada == 1
  StatusPulgus=2
  Pedido.visible=0
ENDIF

IF MovSaida == 1
  StatusPulgus=3
  Pedido.visible=0
ENDIF

IF (MovUsinagem == 0 AND MovEntrada == 0 AND MovSaida == 0)
  StatusPulgus=0
  Pedido.visible=1
ENDIF
//-------------------------------------------------------------------------------------------------------
```

D4. *SCRIPT* DE PROGRAMAÇÃO DO GERENCIADOR DO ARMAZÉM E DA FMC

//Gerenciador_Pulgus.Tags.Bloco2.s_scanner3_robot.retorno_robot_mesa1 (OnValueChanged)

```
IF retorno_robot_mesa1==1

Aplicação.Telas.Movimentação.Usinagem.visible=1

 IF esp_ret_robot_mesa1==1 //carga retornou da usinagem

   tab05_usinagem.Requery()
   tab05_usinagem.SQLQuery("","")

   IF tab05_usinagem.Find("pk_numpallet=" + Str(pallet_mesa11),1)==1
    IF tab05_usinagem.status==16

      tab05_usinagem.BeginTrans()
       tab05_usinagem.status=18
      tab05_usinagem.CommitTrans()

      //grava log
      tab08_log.BeginTrans()
      tab08_log.numpallet=tab05_usinagem.pk_numpallet
      tab08_log.operacao="USI"
      tab08_log.descricao="Registro atualizado na TAB05 status=18"
      tab08_log.data=GetTime()
      IF tab08_log.AddRecord()
       tab08_log.CommitTrans()
      ELSE
       MessageBox("Não foi possível inserir o registro na TAB08")
       tab08_log.RollBack()
      ENDIF

      tab03_armario.Requery()
      tab03_armario.SQLQuery("","")

      IF tab03_armario.Find("tab03_armario.numpallet=" + Str(tab05_usinagem.pk_numpallet),1)==1

        // altera código do produto no retorno da usinagem
```

```
      tab03_armario.BeginTrans()
       tab03_armario.fk_coditem=tab05_usinagem.produto
      tab03_armario.CommitTrans()

      //grava log
      tab08_log.BeginTrans()
      tab08_log.numpallet=tab03_armario.numpallet
      tab08_log.operacao="USI"
      tab08_log.descricao="Registro atualizado na TAB03 fk_coditem=" + tab05_usinagem.produto
      tab08_log.data=GetTime()
      IF tab08_log.AddRecord()
       tab08_log.CommitTrans()
      ELSE
       MessageBox("Não foi possível inserir o registro na TAB08")
       tab08_log.RollBack()
      ENDIF
     ENDIF

    //envio comando paro o transp.-mover pallet até saída usinagem
    saida_usinagem=1
   ELSE
    MessageBox("Pallet não executou o procedimento esperado!")
   ENDIF
  ENDIF
  esp_ret_robot_mesa1=0
  codigo_retorno_M1=0  //zera o código de retorno para mesa1
 ENDIF
ENDIF
//-------------------------------------------------------------------------------------------------------------
```

//Gerenciador_Pulgus.Tags.Bloco2.s_scanner2.presenca_mesa1 (OnValueChanged)

```
Dim var_pal_rec_clp
Dim var_ber_rec_clp


IF presenca_mesa1==1

 IF end_mesa1==1//carga chegou mesa1

  var_pal_rec_clp=Str(Nro_pallet_rec_clp)
  var_ber_rec_clp=Nro_bercousi_rec_clp

 //verifico se o pallet que chegou é o esperado
 IF (var_ber_rec_clp <> 1) And (var_pal_rec_clp <> pallet_mesa11)
  MessageBox("Pallet não esperado na mesa1!Verifique.")
  RETURN
 ELSE
  Nro_bercousi_rec_clp=0
  Nro_pallet_rec_clp=0
 ENDIF

 tab05_usinagem.Requery()
 tab05_usinagem.SQLQuery("","")

 IF tab05_usinagem.Find("pk_numpallet=" + Str(pallet_mesa11),1)
```

```
   pallet_mesa1=tab05_usinagem.pk_numpallet
   IF tab05_usinagem.status==14
    IF tab05_usinagem.mesa==1

      tab05_usinagem.BeginTrans()
       tab05_usinagem.status=16
      tab05_usinagem.CommitTrans()

      //Atualiza a tag ProgUsinagem com o código de usinagem
      ProgUsinagem=tab05_usinagem.produto

      //Atualiza a tag codigo_retorno_M1 (torno=1, fresa=2 e r1=3) + 0 (fase inicial da usinagem) xxx
      IF (ProgUsinagem == "PBR" OR  ProgUsinagem == "PPR" OR ProgUsinagem == "DBR" OR
ProgUsinagem == "DPR")
        codigo_retorno_M1=10
      ELSEIF (ProgUsinagem == "CBR" OR  ProgUsinagem == "CPR")
        codigo_retorno_M1=20
      ELSEIF (ProgUsinagem == "M1R1")
        codigo_retorno_M1=30
      ENDIF

      //indica que vai haver retorno do robô para mesa1
      esp_ret_robot_mesa1=1

      //grava log
      tab08_log.BeginTrans()
      tab08_log.numpallet=pallet_mesa11
      tab08_log.operacao="USI"
      tab08_log.descricao="Pallet chegou na mesa 1"
      tab08_log.data=GetTime()
      IF tab08_log.AddRecord()
       tab08_log.CommitTrans()
      ELSE
       MessageBox("Não foi possível inserir o registro na TAB08")
       tab08_log.RollBack()
      ENDIF
    ENDIF
   ELSE
    MessageBox("Pallet não executou o procedimento esperado!")

    // envia para saída da usinagem
    saida_usinagem=1

   ENDIF
  ELSE
   MessageBox("O processo não foi executado! Pallet não esperado")

   // envia para saída da usinagem
   saida_usinagem=1

  ENDIF
  end_mesa11=0
 ENDIF
ENDIF
//-------------------------------------------------------------------------------------------------------
```

//Gerenciador_Pulgus.Tag.Pulgus.Fresadora.Sinais_Fresa_ASAR (OnValueChanged)

```
// Script para resetar a tag ProgUsinagem a partir da fresadora

IF (Fresa_OK == 0 AND Fresa_Morsa == 1 AND Fresa_Porta == 0)

  IF codigo_retorno_M1 == 20    //setada volta da usinagem da fresa para mesa1
    IF (ProgUsinagem=="PBR" OR ProgUsinagem=="PPR")
      ProgUsinagem=ProgUsinagem
    ELSEIF (ProgUsinagem=="BBR" OR ProgUsinagem=="BPR")
      ProgUsinagem=ProgUsinagem
    ELSEIF (ProgUsinagem=="DBR" OR ProgUsinagem=="DPR")
      ProgUsinagem=ProgUsinagem
    ELSEIF (ProgUsinagem=="M1R1" OR ProgUsinagem=="M2R1")
      ProgUsinagem=ProgUsinagem
    ELSEIF ProgUsinagem=" "
    ENDIF
    codigo_retorno_M1=2
  ENDIF

  IF codigo_retorno_M2 == 20    //setada volta da usinagem da fresa parA mesa2
    IF (ProgUsinagem=="PBR" OR ProgUsinagem=="PPR")
      ProgUsinagem=ProgUsinagem
    ELSEIF (ProgUsinagem=="BBR" OR ProgUsinagem=="BPR")
      ProgUsinagem=ProgUsinagem
    ELSEIF (ProgUsinagem=="DBR" OR ProgUsinagem=="DPR")
      ProgUsinagem=ProgUsinagem
    ELSEIF (ProgUsinagem=="M1R1" OR ProgUsinagem=="M2R1")
      ProgUsinagem=ProgUsinagem
    ELSEIF ProgUsinagem=" "
    ENDIF
    codigo_retorno_M2=2
  ENDIF

ENDIF
//-----------------------------------------------------------------------------------------------------------
```

//Gerenciador_Pulgus.OPCServer.ServerRemoto.RemotoReq (OnValueChanged)

```
DIM num_registros, i
DIM m_numpallet, m_dt_sist, m_coditem, m_pkcelula
DIM m_destino, m_mesa, m_ala, priori, m_alascan

IF StatusRemoto1 == 1
  IF RemotoReq == 1
     Mensagem="Usuário Remoto1 Solicitando Usinagem"
     Aplicação.Telas.Movimentação.Activate()

     MovUsinagem=1
     MovPeca=ServerRemoto.RemotoPeca
     MovUnidades=ServerRemoto.RemotoUnit
  ENDIF
ENDIF

IF (MovUsinagem == 1 AND RemotoReq == 1)
```

```
  IF MovPeca == " " // Se não especificar o código da peça resultante da usinagem
    MessageBox("Para executar um processo de usinagem, é necessário especificar o código da peça
resultante.","Movimentação", 0030h)
    return
  ENDIF

  locate = ("pk_produto = '" + MovPeca + "'")

  tab07_processo.Requery()
  tab07_processo.SQLQuery("", "")

  IF not tab07_processo.Find(locate, 1) // Se o processo de usinagem da peça não existir no cadastro
    MessageBox("Processo de usinagem não cadastrado para a peça " + MovPeca + ".","Movimentação", 0030h)
    return
  ENDIF

  m_coditem = tab07_processo.mat_prima
  m_mesa = tab07_processo.mesa

  locate = ("fk_coditem = '" + m_coditem + "' and status = " + Str(ATIVO))

  tab03_armario.Requery()
  num_registros = tab03_armario.SQLQuery(locate, "")

  IF num_registros == 0 // Se não existir matéria prima disponível para a usinagem
    MessageBox("Processo de usinagem recusado. Para obter a peça " + MovPeca + " é necessário a usinagem
da peça " + m_coditem + " que não se encontra disponível no armazém.","Movimentação", 0030h)
    return
  ENDIF

  IF num_registros < MovUnidades
    IF MessageBox("Quantidade requerida indisponível. É possível produzir apenas " + Str(num_registros)+ "
unidades da peça " + MovPeca + ". Deseja continuar?","Movimentação", 0024h) == 7
      return
    ENDIF
  ELSE
    // Se a qtde disponível for menor que a qtde requerida e for solicitada toda a quantidade disponível
    num_registros = MovUnidades
  ENDIF

  //Grava no log a solicitação da usinagem
  tab08_log.BeginTrans()
  tab08_log.numpallet = 0
  tab08_log.operacao = "USI"
  tab08_log.descricao = "Ordem de usinagem de " + Str(MovUnidades) + " unidades da peça " + MovPeca + ".
Quantidade que será usinada" + Str(num_registros) + "unidades. Estratégia tipo:" + Str(Estrategia)
  tab08_log.data = currentTime
  tab08_log.AddRecord()
  tab08_log.CommitTrans()

  IF MovUrgente == 1
    priori = "U"
  ELSE
    priori = "N"
  ENDIF

  // Loop que gera ordem de movimentação para cada peça a ser usinada
  FOR i = 1 TO num_registros
    locate = ("fk_coditem = '" + m_coditem + "' and status = " + Str(OCUPADO))
    tab03_armario.Requery()
```

```
tab03_armario.SQLQuery(locate, "")

tab03_armario.Find(locate, 1)
m_numpallet = tab03_armario.numpallet
m_dt_sist = tab03_armario.dt_sist
m_alascan = tab03_armario.ala

tab03_armario.MoveFirst()

WHILE not tab03_armario.IsEOF()
  IF (tab03_armario.fk_coditem == m_coditem) // Se o código da peça encontrado for igual ao solicitado
    IF Estrategia == 1  // Se a estratégia for FIFO
      IF tab03_armario.dt_sist < m_dt_sist // Seleciona o pallet de data mais antiga
        m_numpallet = tab03_armario.numpallet
        m_dt_sist = tab03_armario.dt_sist
      ENDIF
    ELSE
      IF Estrategia == 2 // Se a estratégia for LIFO
        IF tab03_armario.dt_sist > m_dt_sist // Seleciona o pallet de data mais recente
          m_numpallet = tab03_armario.numpallet
          m_dt_sist = tab03_armario.dt_sist
        ENDIF
      ELSE // Se a estratégia for a melhor diagonal
        IF tab03_armario.ala < m_alascan // Seleciona o pallet mais próximo da usinagem
          m_numpallet = tab03_armario.numpallet
          m_alascan = tab03_armario.ala
        ENDIF
      ENDIF
    ENDIF
  ENDIF
  tab03_armario.MoveNext()
WEND

tab03_armario.Find("numpallet = " + Str(m_numpallet), 1)

// Seta o pallet para reservado
tab03_armario.BeginTrans()
tab03_armario.status = RESERVADO
tab03_armario.usuario = ServerRemoto.RemotoUser
tab03_armario.CommitTrans()

m_numpallet = tab03_armario.numpallet
m_pkcelula = tab03_armario.pk_celula
m_coditem = tab03_armario.fk_coditem
m_ala = tab03_armario.ala

IF MovRetSaida == 1
  m_destino  = "S01" // berco de saida
ELSE
  m_destino = m_pkcelula
ENDIF

// Gera a ordem de usinagem para a peça em questão
tab05_usinagem.BeginTrans()
tab05_usinagem.pk_numpallet = m_numpallet
tab05_usinagem.produto = MovPeca
tab05_usinagem.mat_prima = m_coditem
tab05_usinagem.mesa = m_mesa
tab05_usinagem.retorno = m_destino
tab05_usinagem.status = 11 // associado
```

```
    tab05_usinagem.AddRecord()
    tab05_usinagem.CommitTrans()

    // Grava a inserção da operação de usinagem na lista de tarefas no arquivo de log
    tab08_log.BeginTrans()
    tab08_log.numpallet = m_numpallet
    tab08_log.operacao = "USI"
    tab08_log.descricao = "Registro inserido na Tab05_Usinagem: pallet " + Str(m_numpallet) + " produto " +
MovPeca + " mat_prima " + m_coditem + " mesa "  + Str(m_mesa) + " retorno " + m_destino + " status 12"
    tab08_log.data = currentTime
    tab08_log.AddRecord()
    tab08_log.CommitTrans()

    // Insere a usinagem do pallet na lista de tarefas
    tab02_tarefas.BeginTrans()
    tab02_tarefas.prioridade = priori
    tab02_tarefas.operacao = "USI"
    tab02_tarefas.numpallet = m_numpallet
    tab02_tarefas.fk_celulaorigem = m_pkcelula
    tab02_tarefas.fk_coditem = m_coditem
    fk_celuladestino  = "S02"
    tab02_tarefas.ala = m_ala
    tab02_tarefas.status = ATIVO
    tab02_tarefas.AddRecord()
    tab02_tarefas.CommitTrans()

    // Grava no log a inserção da tarefa de usinagem
    tab08_log.BeginTrans()
    tab08_log.numpallet = m_numpallet
    tab08_log.operacao = "USI"
    tab08_log.descricao = "Tarefa gerada: USI " + priori + " peça " + m_coditem + " pallet " + Str(m_numpallet)
+ "cel. origem " + m_pkcelula + " cel. destino S02"
    tab08_log.data = currentTime
    tab08_log.AddRecord()
    tab08_log.CommitTrans()

    Mensagem="Pedido Inserido com sucesso!"

  NEXT
ENDIF

RetSaida.visible = 0
MovSaida = 0
MovEntrada = 0
MovUsinagem = 0
MovUrgente = 0
MovPallet = 0
MovPeca = " "
MovCelula = " "
MovUnidades = 1

DescUnidades.visible = 1
DescPallet.visible = 0
DescCelula.visible = 0
//-------------------------------------------------------------------------------------------------------------
```

//Gerenciador_Pulgus.OPCServer.ServerFresadora.ComunicFresa_Robo (OnValueChanged)

```
Sinais_Fresa_ASAR=ComunicFresa_Robo
//-------------------------------------------------------------------------------------------------------------
```

//Gerenciador_Pulgus.OPCServer.ServerR1.ComunicR1_Robo (OnValueChanged)

```
Sinais_R1_ASAR=ComunicR1_Robo
//-----------------------------------------------------------------------------------------------------------
```

//Gerenciador_Pulgus.OPCServer.ServerTorno.ComunicTorno_Robo (OnValueChanged)

```
Sinais_Torno_ASAR=ComunicTorno_Robo
//-----------------------------------------------------------------------------------------------------------
```

//Gerenciador_Pulgus.OPCServer.ServerRobo.SINAIS_DO_ROBO (OnValueChanged)

```
Sinais_Robo_ASAR=SINAIS_DO_ROBO
//-----------------------------------------------------------------------------------------------------------
```

//Gerenciador_Pulgus.OPCServer.ServerRemoto.StatusRemoto1 (OnValueChanged)

```
IF StatusRemoto1 == 1
  Mensagem="Usuário Remoto1 Conectado"
ELSEIF StatusRemoto1 == 0
  Mensagem=" "
ENDIF
//-----------------------------------------------------------------------------------------------------------
```

//Gerenciador_Pulgus.Pulgus.Robo.Robo_M1 (OnValueChanged)

```
IF Robo_M1==1
 retorno_robot_mesa1=1
ELSEIF Robo_M1==0
 retorno_robot_mesa1=0
ENDIF
//-----------------------------------------------------------------------------------------------------------
```

//Gerenciador_Pulgus.Pulgus.Robo.Robo_M2 (OnValueChanged)

```
IF Robo_M2==1
 retorno_robot_mesa2=1
ELSEIF Robo_M2==0
 retorno_robot_mesa2=0
ENDIF
//-----------------------------------------------------------------------------------------------------------
```

ANEXO EM CD-ROM