



## Campus de São Carlos

METAMODELO PARA A MODELAGEM E  
SIMULAÇÃO DE SISTEMAS A EVENTOS  
DISCRETOS, BASEADO EM REDES DE PETRI E  
REALIDADE VIRTUAL. UMA APLICAÇÃO EM  
SISTEMA DE MANUFATURA.

**JANDIRA GUENKA PALMA**

Orientador: Prof. Assoc. Arthur José Vieira Porto

**UNIVERSIDADE DE SÃO PAULO**



**ESCOLA DE ENGENHARIA  
DE SÃO CARLOS**

METAMODELO PARA A  
MODELAGEM E SIMULAÇÃO DE SISTEMAS A  
EVENTOS DISCRETOS, BASEADO EM REDES DE  
PETRI E REALIDADE VIRTUAL. UMA APLICAÇÃO  
EM SISTEMA DE MANUFATURA.

JANDIRA GUENKA PALMA

DEDALUS - Acervo - EESC



31100036866

Serviço de Pós-Graduação EESC/USP

EXEMPLAR REVISADO

Data de entrada no Serviço: 09/03/06

*Juci Oliveira*

Ass.: *Juci Oliveira*



Tese apresentada à Escola de Engenharia de São Carlos, da Universidade de São Paulo, como parte dos requisitos para obtenção do Título de Doutor em Engenharia Mecânica.

ORIENTADOR: Prof. Dr. Arthur José Vieira Porto

São Carlos  
2001

Class. TESE-EESC  
Cutt. 16/3991  
Tombo T0049/02

20 1032004

Ficha catalográfica preparada pela Seção de Tratamento  
da Informação do Serviço de Biblioteca – EESC/USP

P171m

Palma, Jandira Guenka

Metamodelo para a modelagem e simulação de sistemas  
a eventos discretos, baseado em redes de petri e  
realidade virtual : uma aplicação em sistema de  
manufatura / Jandira Guenka Palma. -- São Carlos,  
2001.

Tese (Doutorado) -- Escola de Engenharia de São  
Carlos-Universidade de São Paulo, 2001.

Área: Engenharia Mecânica.

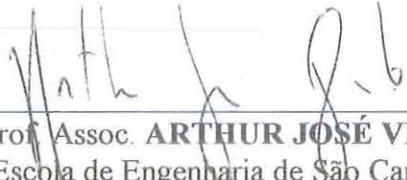
Orientador: Prof. Dr. Arthur José Vieira Porto.

1. Redes de petri. 2. Ambientes virtuais.
3. Sistemas de eventos discretos. 4. Simulação.
- I. Título.

## FOLHA DE JULGAMENTO

Candidata: Bacharela **JANDIRA GUENKA PALMA**

Tese defendida e julgada em 14-12-2001 perante a Comissão Julgadora:

  
Prof. Assoc. **ARTHUR JOSÉ VIEIRA PORTO** (Orientador)  
(Escola de Engenharia de São Carlos/USP)

APROVADA

  
Prof. Assoc. **EDUARDO VILA GONÇALVES FILHO**  
(Escola de Engenharia de São Carlos/USP)

APROVADA

  
Prof. Dr. **MARCOS RIBEIRO PEREIRA BARRETO**  
(Escola Politécnica/USP)

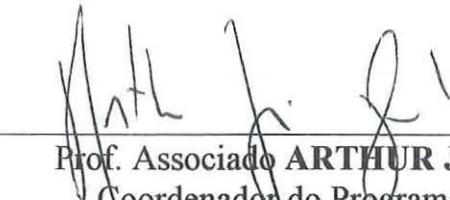
APROVADA

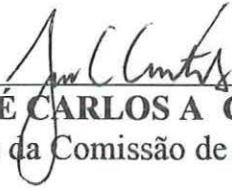
  
Prof. Dr. **JOÃO UMBERTO FURQUIM DE SOUZA**  
(Universidade Estadual de Ponta Grossa/UEPG)

APROVADA

  
Prof. Dr. **DARIO HENRIQUE ALLIPRANDINI**  
(Universidade Federal de São Carlos/UFSCar)

Aprovado

  
Prof. Associado **ARTHUR JOSÉ VIEIRA PORTO**  
Coordenador do Programa de Pós-Graduação  
em Engenharia Mecânica

  
**JOSE CARLOS A CINTRA**  
Presidente da Comissão de Pós-Graduação

*Ao meu marido,*

*Ricardo*

## **Agradecimentos**

Ao meu orientador, Prof. Dr. Arthur José Vieira Porto, pela orientação, pelas contribuições na realização do trabalho, pelo incentivo e amizade.

Ao Jander pelas importantes contribuições na realização deste trabalho;

Daniel, Tiago Tel e Fábio pelas contribuições na realização deste trabalho.

Ao meu marido, Ricardo, pelo apoio, pelo incentivo e pelas contribuições na revisão deste trabalho.

Daniel Kaster, Patrícia, Sayuri e pelas contribuições na revisão deste trabalho.

Aos colegas de trabalho Ana Paula, Andrea, Anna Cristina, Beth, Carlos Ravelli, Edílson, Giovani, Helena, José Luiz, Lobão, Mamoru, Marcelo, Mariella, Mário Tronco, Orides, Osvaldo Assato, Osvaldo Oshiro, Patrícia, Politano, Renato, Ricardo, Roberto.

A Profa. Cleusa Asanome pelo apoio e incentivo ao trabalho.

A todos professores e funcionários do departamento na ajuda e a amizade.

Ao CNPq, pela bolsa concedida.

# Sumário

LISTA DE FIGURAS .....	V
LISTA DE TABELAS .....	VII
LISTA DE ABREVIATURAS E SIGLAS.....	VIII
LISTA DE SÍMBOLOS .....	IX
RESUMO.....	X
ABSTRACT.....	XI
<b>1. INTRODUÇÃO .....</b>	<b>1</b>
1.1 CONTEXTO.....	2
1.2 OBJETIVO .....	3
1.3 JUSTIFICATIVA.....	4
1.4 ORGANIZAÇÃO DO TRABALHO .....	6
<b>2. REVISÃO BIBLIOGRÁFICA .....</b>	<b>8</b>
2.1 INTRODUÇÃO.....	8
2.2 CONSIDERAÇÕES INICIAIS DA SIMULAÇÃO .....	8
2.3 SIMULAÇÃO.....	11
2.3.1 <i>Processo de Simulação</i> .....	12
2.3.2 <i>Classificação da Simulação</i> .....	12
2.3.2.1 Simulação de Eventos Discretos.....	13
2.3.3 <i>Simulação Computacional</i> .....	14
2.4 MODELAGEM.....	15
2.4.1 <i>Redes de Petri</i> .....	17
2.4.1.1 Definição formal .....	20
2.4.1.2 Comportamento Dinâmico da RP .....	21
2.4.1.3 Extensões de Rede de Petri.....	22
2.4.1.4 Softwares para Redes de Petri.....	23
2.5 SISTEMAS DE MANUFATURA DISCRETAS .....	25
2.5.1 <i>Aplicações de Redes de Petri na Manufatura</i> .....	26

2.5.2	<i>Visualização da Simulação de Sistemas de Manufatura</i> .....	26
2.6	<b>REALIDADE VIRTUAL</b> .....	27
2.6.1	<i>Dispositivos de Realidade Virtual</i> .....	29
2.6.2	<i>Conceitos de Imersão, Iteração e Envolvimento</i> .....	31
2.6.3	<i>RV Passiva, Exploratória ou Iterativa</i> .....	32
2.6.3.1	<i>Tipos de Realidade Virtual</i> .....	33
2.6.4	<i>Realidade Virtual na Manufatura</i> .....	34
2.6.4.1	<i>Visão geral sobre o emprego da realidade virtual na indústria</i> .....	35
2.6.5	<i>Realidade Virtual e Simulação</i> .....	38
2.6.6	<i>Ambientes especializados de realidade virtual</i> .....	40
2.7	<b>MODELAGEM E ANÁLISE DE AVs COM RP</b> .....	42
2.8	<b>ENGENHARIA DE SOFTWARE - ORIENTAÇÃO A OBJETOS</b> .....	44
2.8.1	<i>Análise Orientada a Objeto</i> .....	45
2.8.2	<i>Conceitos de Orientação a Objetos</i> .....	47
2.8.2.1	<i>Abstração</i> .....	47
2.8.2.2	<i>Encapsulamento ou Proteção de Informação</i> .....	48
2.8.2.3	<i>Herança</i> .....	48
2.8.2.4	<i>Polimorfismo</i> .....	48
2.8.2.5	<i>Agregação</i> .....	48
2.8.3	<i>Linguagens de Programação Orientada a Objetos</i> .....	49
2.8.4	<i>O paradigma de Orientação Objeto em Sistemas de Manufatura</i> .....	52
<b>3.</b>	<b>PROPOSTA DE UM METAMODELO PARA MODELAGEM E SIMULAÇÃO DE SISTEMA A EVENTOS DISCRETOS, BASEADO EM REDES DE PETRI E REALIDADE VIRTUAL</b> .....	54
3.1	<b>INTRODUÇÃO</b> .....	54
3.2	<b>MODELAGEM E SIMULAÇÃO BASEADA EM REDES DE PETRI</b> .....	55
3.3	<b>INTERFACE DE RV PARA A SIMULAÇÃO</b> .....	57
3.4	<b>METAMODELO</b> .....	60
3.4.1	<i>Módulo de Edição e Validação de Modelos - MEVM</i> .....	64
3.4.1.1	<i>Recomendações para a elaboração de modelos para controle de AVs.</i> .....	65
3.4.2	<i>Ferramenta de Desenvolvimento de Classes - FDC</i> .....	69
3.4.2.1	<i>Caracterização das Classes dos Objetos Virtuais</i> .....	70
3.4.2.2	<i>Programação das Classes</i> .....	71
3.4.3	<i>Editor de Ambientes Virtuais - EAV</i> .....	74
3.4.4	<i>Módulo de Interligação - MI</i> .....	77

3.4.5 <i>Ferramentas para Executar a Simulação e Gerenciar a Comunicação</i>	79
3.4.5.1 <i>Simulador do Ambiente Virtual -SAV</i> .....	82
3.4.6 <i>Módulo Central de Controle - MCC</i> .....	85
3.5 <i>CONSIDERAÇÕES FINAIS DO METAMODELO</i> .....	88
<b>4. APLICAÇÃO DO METAMODELO PARA A IMPLEMENTAÇÃO DE UM SISTEMA COMPUTACIONAL DE SIMULAÇÃO COM INTERFACE DE REALIDADE VIRTUAL PARA SISTEMAS DE MANUFATURA.....</b>	<b>89</b>
4.1 <i>INTRODUÇÃO</i> .....	89
4.2 <i>RECURSOS FÍSICOS</i> .....	90
4.3 <i>IMPLEMENTAÇÃO</i> .....	90
4.3.1 <i>Edição do Modelo e a Validação</i> .....	90
4.3.1.1 <i>Interpretação do sistema de armazenamento do Editor</i> .....	91
4.3.2 <i>Criação das Classes dos Objetos Virtuais</i> .....	92
4.3.3 <i>Editor de Ambientes Virtuais</i> .....	96
4.3.4 <i>Módulo de Interligação – MI</i> .....	98
4.3.5 <i>Simulação com Interface de RV</i> .....	100
4.3.5.1 <i>Simulador de Ambientes Virtuais</i> .....	102
4.3.5.2 <i>Sistema de Controle</i> .....	103
4.4 <i>DESENVOLVIMENTO DE UM SISTEMA DE SIMULAÇÃO DE MANUFATURA COM INTERFACE DE RV</i> .....	107
4.4.1 <i>Concepção ou formulação do modelo</i> .....	108
4.4.2 <i>Implementação e validação do modelo</i> .....	108
4.4.3 <i>Edição da Interface gráfica para o modelo</i> .....	108
4.4.4 <i>Integração do modelo com AV</i> .....	108
4.4.5 <i>Execução</i> .....	109
4.5 <i>ESTUDO DE CASO</i> .....	109
4.5.1 <i>Passo 1 - Criar o modelo</i> .....	109
4.5.2 <i>Passo 2 – Editar o modelo e validá-lo</i> .....	111
4.5.3 <i>Passo 3 - Edição do Ambiente Virtual</i> .....	114
4.5.4 <i>Passo 4 – Integrar o Modelo com o Ambiente Virtual</i> .....	114
4.5.5 <i>Passo 5 – Executar a Simulação</i> .....	116
4.5.6 <i>Observações</i> .....	117
<b>5. CONCLUSÕES .....</b>	<b>118</b>
5.1 <i>TRABALHOS FUTUROS</i> .....	119
<b>6. BIBLIOGRAFIA.....</b>	<b>121</b>

<b>APÊNDICE A - PROPRIEDADES E MÉTODOS DE ANÁLISE DA REDE DE PETRI .....</b>	<b>A-1</b>
<b>APÊNDICE B - FERRAMENTAS DE REDES DE PETRI.....</b>	<b>B-1</b>
<b>APÊNDICE C - CLASSES DO SISTEMA DE CONTROLE.....</b>	<b>C-1</b>

## Lista de Figuras

<b>Figura 1.</b>	<b>Passos de utilização dos Sistemas Comerciais de Simulação.....</b>	<b>2</b>
<b>Figura 2.</b>	<b>Simulação com Interface em Realidade Virtual.....</b>	<b>3</b>
<b>Figura 3.</b>	<b>Elementos da Rede de Petri .....</b>	<b>18</b>
<b>Figura 4.</b>	<b>Fórmula química (<math>2\text{H}_2 + \text{O}_2 \rightarrow 2\text{H}_2\text{O}</math>) modelada em RP.....</b>	<b>19</b>
<b>Figura 5.</b>	<b>Composição dos Sistemas de Manufatura.....</b>	<b>25</b>
<b>Figura 6.</b>	<b>Realidade Virtual .....</b>	<b>28</b>
<b>Figura 7.</b>	<b>Vistas da fábrica de engrenagens virtual, .....</b>	<b>39</b>
<b>Figura 8.</b>	<b>Esquema de obtenção das rotinas para o ambiente virtual do trabalho .....</b>	<b>43</b>
<b>Figura 9.</b>	<b>Todo-Parte .....</b>	<b>49</b>
<b>Figura 10.</b>	<b>Componentes básicos de LPOO.....</b>	<b>52</b>
<b>Figura 11.</b>	<b>Ambientes do Metamodelo.....</b>	<b>60</b>
<b>Figura 12.</b>	<b>Metamodelo para a Modelagem e Simulação de Eventos Discretos com Interface de Realidade Virtual.....</b>	<b>61</b>
<b>Figura 13.</b>	<b>Modelo de um sistema com 2 Agvs para servir uma Máquina.....</b>	<b>67</b>
<b>Figura 14.</b>	<b>Exemplo do framework .....</b>	<b>76</b>
<b>Figura 15.</b>	<b>Tabela de Conexão realiza a conexão do modelo com AV.....</b>	<b>78</b>
<b>Figura 16.</b>	<b>Atividades do Módulo de Central de Controle – MCC e do Simulador de Ambiente Virtual - SAV.....</b>	<b>81</b>
<b>Figura 17.</b>	<b>Configuração do Simulador de Ambientes Virtuais.....</b>	<b>83</b>
<b>Figura 18.</b>	<b>Monitoramento do Ambiente Virtual.....</b>	<b>84</b>
<b>Figura 19.</b>	<b>Módulo Central de Controle.....</b>	<b>85</b>
<b>Figura 20.</b>	<b>Diagrama de Estado do MCC.....</b>	<b>86</b>
<b>Figura 21.</b>	<b>Estado Inicial (a), Estado (b), Estado Final (c). .....</b>	<b>87</b>
<b>Figura 22.</b>	<b>Transição de Estado .....</b>	<b>87</b>
<b>Figura 23.</b>	<b>Editor PetriNetTools.....</b>	<b>91</b>
<b>Figura 24.</b>	<b>Classes dos Objetos Virtuais .....</b>	<b>94</b>
<b>Figura 25.</b>	<b>Desenvolvimento do modelo geométrico 3D .....</b>	<b>94</b>
<b>Figura 26.</b>	<b>Arquivo de configuração .....</b>	<b>96</b>
<b>Figura 27.</b>	<b>Editor de Ambientes Virtuais.....</b>	<b>97</b>
<b>Figura 28.</b>	<b>Servidor de Simulação - Entrada de Dados.....</b>	<b>99</b>
<b>Figura 29.</b>	<b>Servidor de Simulação - Módulo de Interligação: Associação .....</b>	<b>99</b>

<b>Figura 30. Servidor de Simulação – Clientes Conectados e Início da Execução da Simulação.....</b>	<b>100</b>
<b>Figura 31. Comunicação do Servidor/Cliente .....</b>	<b>101</b>
<b>Figura 32. Arquivo gerado por um editor de Redes de Petri .....</b>	<b>105</b>
<b>Figura 33. Rota dos AGVs modelado em Redes de Petri.....</b>	<b>110</b>
<b>Figura 34. Janela de simulação .....</b>	<b>112</b>
<b>Figura 35. Propriedades e Análise da RP.....</b>	<b>113</b>
<b>Figura 36. Arquivo do Ambiente Virtual gerado pelo Editor .....</b>	<b>114</b>
<b>Figura 37. Módulo de interligação – Parâmetros dos Métodos .....</b>	<b>115</b>
<b>Figura 38. Tabela de conexão gerado pelo MI .....</b>	<b>115</b>
<b>Figura 39. Ambiente Virtual.....</b>	<b>117</b>

## Lista de Tabelas

<b>Tabela 1.</b> Soluções para as dificuldades da simulação,.....	10
<b>Tabela 2.</b> Interpretações típicas de Transições e Lugares.....	18
<b>Tabela 3.</b> Interpretação da fórmula química ( $2\text{H}_2 + \text{O}_2 \rightarrow 2\text{H}_2\text{O}$ ).....	19
<b>Tabela 4.</b> Diferença da simulação convencional com a simulação com interface de RV.....	57
<b>Tabela 5.</b> Comparação do Desenvolvimento AV com a Proposta.....	59
<b>Tabela 6.</b> Pontos relevantes do comportamento dos modelos de RP e dos ambientes de RV para o metamodelo.....	63
<b>Tabela 7.</b> Exemplos de tipos enumerados.....	74

## Lista de Abreviaturas e Siglas

AEE	- Advanced Engineering Environment
AGV	- Auto-Guided Vehicle (Veículo Auto-Guiado)
AV	- Ambientes Virtuais
BCOV	- Biblioteca de Classes de Objetos Virtuais
CIM	- <i>Computer Integrated Manufactury</i> (Manufatura Integrada por Computador)
DFD	- Diagrama de Fluxo de Dados
EAV	- Editor de Ambientes Virtuais
FDC	- Ferramenta de Desenvolvimento de Classes
FMS	- <i>Flexible Manufacturing System</i> (Sistemas Flexíveis de Manufatura)
MCC	- Módulo Central de Controle
MEVM	- Módulo de Edição e Validação de Modelos
MI	- Módulo de Interligação
MI	- Módulo de Interligação
OO	- Orientado a Objeto
OV	- Objeto Virtual
POO	- Programação Orientada a Objetos
RP	- Redes de Petri ou PN – <i>Petri Net</i>
RA	- Realidade Aumentada -( <i>Aumented Reality</i> )
RM	- Realidade Melhorada ( <i>Enhanced Reality</i> ).
RV	- Realidade Virtual
SAV	- Simulador de Ambientes Virtuais
SAV	- Simulador de Ambientes Virtuais
SED	- Sistema de Eventos Discretos
WBS	- <i>Work Breakdown Structure</i>

## Lista de Símbolos

- $A$  - a matriz de incidência
- $d_{ij}$  - a distância sincrônica entre duas transições ( $i$  e  $j$ )
- $F$  - um conjunto de arcos (relação de fluxo)
- $K$  - a função capacidade
- $L(M_0)$  - o conjunto de todas possíveis seqüências de disparos de transições a partir de  $M_0$
- $p$  - o conjunto de transições de entrada de  $p$
  - $p$  • - o conjunto de transições de saída de  $p$
- $P$  - um conjunto finito de lugares
- $t$  - o conjunto de lugares de entrada de  $t$
  - $t$  • - o conjunto de lugares de saída de  $t$

## RESUMO

PALMA, J. G. (2001). *Metamodelo para modelagem e simulação de sistemas a eventos discretos, baseado em Redes de Petri e Realidade Virtual. Uma aplicação em sistemas de manufatura.* São Carlos, 2001. Tese (Doutorado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.

Uma vez que uma aplicação ou projeto tenha sido identificado como sujeito ao uso da simulação, decisões devem ser tomadas acerca de como conduzir os estudos. Embora não haja regras definitivas, alguns passos são geralmente recomendados, tais como: planejamento do estudo, definição do sistema, construção do modelo, execução dos experimentos, análise dos resultados e relatório final. E, a construção do modelo é uma das etapas mais demoradas e complexas.

Na execução dos experimentos, e na análise dos resultados a Realidade Virtual (RV) é uma interface que pode fornecer um suporte importante para a tomada de decisões, pois a RV auxilia na criação de mundos virtuais semelhantes ao mundo real, que ajudam na compreensão do funcionamento dos sistemas.

Este trabalho propõe um metamodelo para o desenvolvimento de sistemas de simulação de eventos discretos com interface de RV aplicados a ambientes ou estações de trabalho de manufatura. A proposta do metamodelo é composto por quatro módulos: i) de edição e validação de modelos, ii) de criação, edição e execução de ambientes virtuais, iii) de conexão para efetuar o elo entre os dois primeiros módulos e, iv) de controle para gerenciar a comunicação e controlar a simulação.

Como resultado tem-se a simulação centralizada baseada no modelo descrito em Rede de Petris (RP) com interface de RV distribuída. O sistema computacional gerado pelo metamodelo auxiliará no aprendizado e compreensão do problema simulado, e ainda permitirá ao usuário o envolvimento através da iteração.

**Palavras Chaves:** Simulação, Redes de Petri, Ambiente Virtual, Sistemas de Eventos Discretos.

## ABSTRACT

PALMA, J. G. Metamodel for Modeling and Discrete-Event System Simulation based in Petri Net and Virtual Reality. A Manufacture System application. São Carlos, 2001. p. Tese (Doutorado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.

The VR aids in the creation of virtual worlds similar to the real world, helping in understanding how a system works. This research work is concerned with the development of a discrete-event system simulation based on Petri Nets, with Virtual Reality interface for manufacturing environment or workstations. The system is composed of four modules, the first is model editing and validation, the second module is a modeling tool for virtual objects and/or a library of virtual elements, the third module makes the link between the two previous modules, and finally, the fourth is a simulation and control module. The centralized simulation is based on models described by Petri Nets with distributed RV interface resulting in a flexible and consistent system. The system will help users in learning and understanding the simulation problem, and it will also allow the user's integration with the environment through the interaction, and through distributed interface.

Keywords: Petri Nets, Simulation, Virtual Environment, Discrete-event System

## 1. Introdução

Historicamente, a simulação era usada em novos projetos de manufatura ou renovações de grande porte. Hoje, o aperfeiçoamento desta técnica permite que se otimize pequenos pontos de uma produção e se consiga avançar na integração de operações em uma fábrica, (NORMAN, 1992).

A simulação possibilita maior agilidade na concretização de projetos, bem como a viabilidade de novos estudos, uma vez que permite diminuir os custos, pois analisa alternativas sem utilizar equipamentos reais em todo o estudo. Desta forma, os sistemas de simulação tornam-se fundamentais, pois permitem que novas idéias sejam testadas, ou que sistemas prontos sejam postos à prova para se analisar pontos falhos e corrigi-los com facilidade.

No futuro, as realizações dos negócios na fábrica serão feitas a partir dos resultados obtidos pelos sistemas de simulação, pois as principais decisões sempre necessitarão ser avaliadas em todos os seus aspectos. O uso da modelagem de simulação se estenderá das aplicações tradicionais da manufatura e aspectos logísticos a processos de negócio e a aplicações de simulações iterativas no treinamento e vendas, penetrando em todos os aspectos das corporações e em muitos aspectos das vidas pessoais, (DONALD et al., 1999) e (PORTO&PALMA, 2000).

## 1.1 Contexto

A tecnologia da simulação teve um crescimento rápido desde os seus primórdios nos anos de 1960. Atualmente, muitas áreas confiam no uso intensivo da simulação para testar novas idéias e opções.

Hoje, no mercado existem mais de uma centena de softwares e sistemas de simulação que variam consideravelmente em muitos aspectos – custo, funcionalidade, flexibilidades, facilidade de uso, plataforma. ARENA®, AUTOMOD®, WITNESS®, Taylor II®, SLAM®, ProModel® etc. são alguns exemplos de sistemas comerciais de simulação comercializados no Brasil.

O usuário, projetista de simulação, dos aplicativos relacionados abstrai o problema em forma de um modelo compatível com o software, faz a edição, coordena a execução pela interface gráfica e visualiza a simulação pela animação e os dados produzidos, e repete este processo até terminar de analisar o sistema (Figura 1).

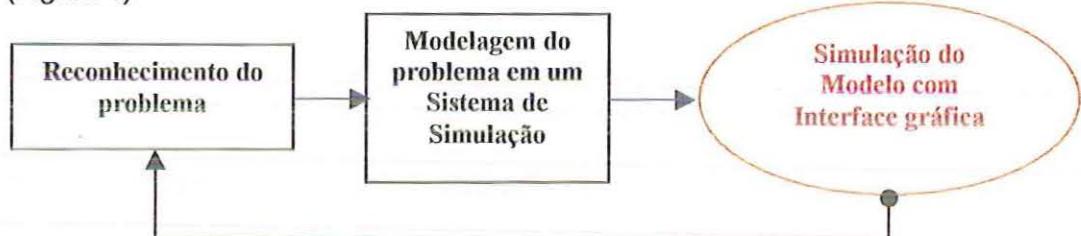
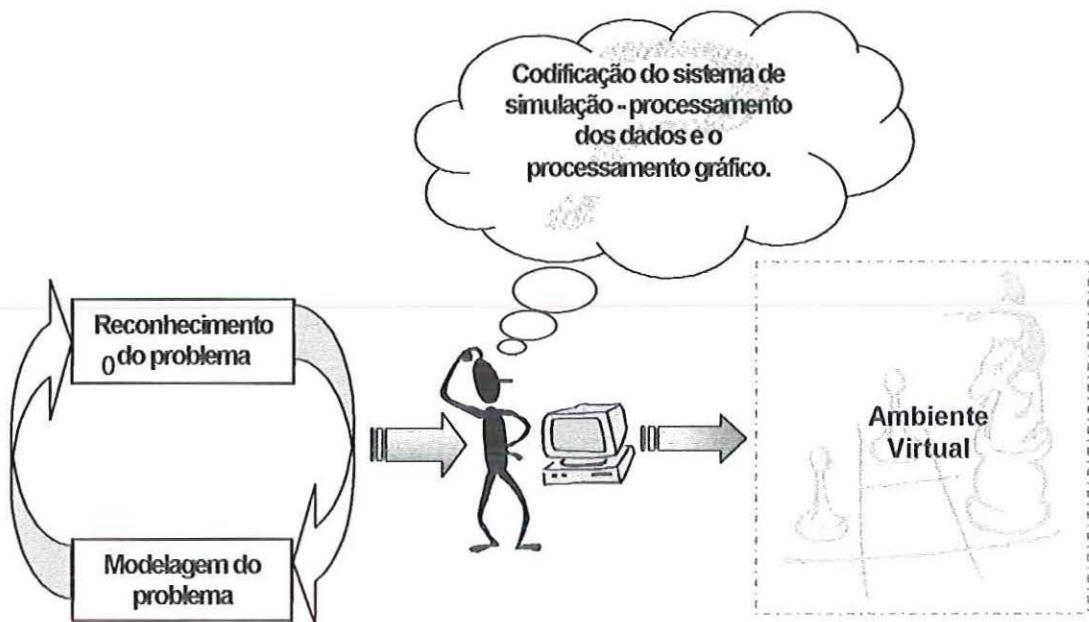


Figura 1. Passos de utilização dos Sistemas Comerciais de Simulação

Os softwares comerciais de simulação apresentam animações gráficas, mas atualmente já existem simulações utilizando Ambientes Virtuais (AV) que oferecem a possibilidade de participação do usuário durante a execução da simulação através da interação e imersão.

COMMITTEE ON ADVANCED ENGINEERING ENVIRONMENTS, NATIONAL RESEARCH COUNCIL (1999); KIRNER(1999) e COMMITTEE TO STUDY INFORMATION TECHNOLOGY AND MANUFACTURING, NATIONAL RESEARCH COUNCIL(1995) relatam simulações em Ambientes Virtuais. Mas os AVs normalmente são programados para fim específico, ou seja, o reconhecimento do problema e a modelagem foram descritos para o analista de sistemas, e este por sua vez gerou o código correspondente ao modelo com interface de Realidade Virtual (RV).

A Figura 2 demonstra os passos normalmente seguidos por estes sistemas.



**Figura 2.** Simulação com Interface em Realidade Virtual

O desenvolvimento de um sistema computacional com interface de RV demanda conhecimentos especializados, como a programação de sistemas em tempo real, orientação a objetos, redes, modelagem geométrica, modelagem física, multitarefa etc. como consequência a construção de ambientes virtuais é onerosa e consome muito tempo de elaboração (KIRNER, 1999).

## 1.2 *Objetivo*

O objetivo deste trabalho é elaborar um Metamodelo para o desenvolvimento de aplicativos de simulação com as características semelhantes aos softwares comerciais citados acima e ainda os aplicativos devem fornecer ao projetista de simulação a possibilidade criar ambientes em realidade virtual em que os usuários finais possam iterar e imergir durante a execução da simulação.

O aplicativo desenvolvido a partir do Metamodelo, tem como finalidade atuar como uma ferramenta para o projetista de simulação desenvolver sistemas de simulação com interfaces de RV de forma rápida e eficiente. A criação de bibliotecas de elementos virtuais de manufatura é uma solução para a construção de ambientes com interface tridimensional, e os editores de Redes de Petri facilitam

a modelagem e a “validação” de um sistema. O desafio é estabelecer a comunicação entre eles (o modelo e a interface) e executar a simulação, de forma dinâmica e com flexibilidade para atender a diversidade dos sistemas de manufatura.

### **1.3 Justificativa**

A necessidade crescente de novas ferramentas para o aumento da produtividade, como reflexo do aumento da competitividade entre as diversas empresas, tem gerado nos últimos anos uma corrida tecnológica sem precedentes históricos. O principal campo científico envolvido nessa corrida é o da computação de forma direta (melhoria dos *hardwares* e *softwares* computacionais) ou indireta (aplicação da computação na automação industrial, na medicina, nos equipamento, na gestão de negócios etc.).

Atualmente, um dos principais ápices da tecnologia é a realidade virtual, mas só recentemente ela se tornou disponível com plataformas a preços acessíveis para a maioria dos engenheiros de simulação, (JAIN, 1999).

A Realidade Virtual é uma opção que pode melhorar a qualidade da interface e aumentar consideravelmente a produtividade. Esta tecnologia emergente mostra-se como a mais “amigável” para a construção de interfaces homem-computador, pois permite às pessoas a iteração e a navegação em um ambiente sintético tridimensional, gerado por computador, e a animação gráfica tridimensional é um excelente meio de estabelecer a credibilidade de um modelo de simulação, (JONES&CYGNUS, 1993).

A tecnologia de RV tem se mostrada vantajosa para processos de simulação convencional de forma que pode auxiliar o entendimento de problemas complexos de engenharia. Um sistema de RV não fornece somente funções computadorizadas básicas, mas também um meio eficiente para utilizar os sentidos humano e a iteração em um ambiente virtual, (PENG, 2000).

O sistema de simulação com interface de Realidade Virtual pode trazer outras vantagens como (BARNERS, 1996):

- a simplificação da apresentação dos resultados para a alta gerência e para os operadores de chão de fábrica;
- um meio de abstração elaborado (representação sintética do ambiente real) que permite a mentalização das idéias com agilidade, facilitando nas especulações das possíveis soluções e tomadas de decisões;
- a ilustração das alternativas do sistema conforme as ações executadas em tempo real, auxiliando os engenheiros e operadores no controle do sistema;
- auxílio no treinamento de supervisores através de ações que podem ser efetuadas no mundo virtual (permitindo a averiguação do comportamento do sistema e influências destas ações antes de utilizá-las no ambiente real).

Sabe-se também que a integração dos diversos meios de comunicação, como som, imagem e texto, gera uma iteração bem mais efetiva que esses diversos meios separadamente. O processo de assimilação e compreensão das informações torna-se muito mais rápido. Geralmente, retemos 20% do que vemos, 30% do que ouvimos, 50% do que vemos e ouvimos, e 80% do que vemos, ouvimos e fazemos, simultaneamente, (REVISTA DE INFORMAÇÃO E TECNOLOGIA, 1997).

Existem várias ferramentas que ajudam construir ambientes virtuais, mas todas dependem de conhecimentos específicos de programação de RV, e por outro lado, há vários sistemas de simulação que possuem animação gráfica 2D ou 3D para o acompanhamento da execução da simulação, mas não permitem a iteração e a imersão em tempo de execução.

Conforme o COMMITTEE ON MODELING AND SIMULATION (1997) está faltando ferramentas de *hardware* e *software* sofisticadas para construir eficientemente ambientes complexos e de grande porte na defesa e indústrias de entretenimento. Um dos autores do livro, Jack Thorpe, declarou que as ferramentas existentes são ardilosas e primitivas e exige um treinamento significativo para dominá-las. Uma melhoria das ferramentas poderia ajudar na redução do tempo e do custo de criação das simulações através da automação de algumas das tarefas

que ainda são feitas manualmente. Alex Seiden, da Industrial Light and Magic, comenta no livro que o software é uma área carente na simulação.

Assim, a proposta do metamodelo (junção de várias ferramentas) é justificada, pois possibilita ao analista de sistemas desenvolver softwares que são aptos a criar ambientes de simulações com interface em realidade virtual, de forma que os usuários destes softwares (projetistas de simulação) utilizem somente conhecimentos de sistemas de simulação e de forma intuitiva.

Portanto, o software produzido a partir do metamodelo possibilitará a criação de ambientes de eventos discretos com interface de Realidade Virtual (RV), sem a utilização das linguagens de programação (linhas de código), deixando-o acessível aos projetistas de simulação de sistemas de manufatura, sem precisar da figura do analista de sistema.

Deste modo, não será somente uma animação, e sim um ambiente digital que representa com certo realismo o problema e reage conforme as ações do usuário em tempo real.

#### **1.4 Organização do Trabalho**

O contexto, a motivação, os objetivos e a justificativa deste trabalho foram apresentados neste capítulo.

O Capítulo 2 apresenta uma revisão bibliográfica dos componentes do metamodelo e das ferramentas necessárias para desenvolver o sistema: Modelagem e Simulação de Sistemas de Manufatura, formalização dos modelos em rede de Petri Lugar/Transição, Realidade Virtual, Simulação com Realidade Virtual e sobre o paradigma de Programação Orientada Objetos.

O Capítulo 3 apresenta a proposta do Metamodelo, abordando a arquitetura e aspectos operacionais.

O Capítulo 4 apresenta a implementação de um sistema baseado no metamodelo proposto e um estudo de caso.

O Capítulo 5 apresenta as conclusões.

## 2. Revisão Bibliográfica

### 2.1 Introdução

A proposta do metamodelo é fornecer um caminho para o desenvolvimento de sistemas de simulação com interface de Realidade Virtual (RV), para alcançar esta meta serão englobadas várias áreas do conhecimento, tais como: Simulação, modelagem de sistemas de manufatura; modelagem lógica em Redes de Petri (RP); criação e manipulação de ambientes virtuais; e o paradigma de orientação a objeto.

### 2.2 Considerações Iniciais da Simulação

O mundo já passou por várias fases, agrícola, mercantil, industrial e hoje o que prevalece é a informação. A utilização e a manipulação das informações tem proporcionado muitos avanços, mas muitos desses poderiam ser maximizados com a inclusão da tecnologia de simulação existente. Entretanto, estas oportunidades estão sendo perdidas ou a tecnologia é reinventada pela falta de comunicação, *marketing*, e convergência de interesses, (PORTO & PALMA, 2000) .

Conforme BANKS (1999) a simulação é uma ferramenta indispensável na solução de muitos problemas quando usada para descrever e analisar o comportamento de um sistema real, além de responder perguntas sobre este sistema.

BANKS et al. (1995) mostram várias vantagens da aplicação da simulação na manufatura como as listadas abaixo:

- Novas políticas, procedimentos operacionais, regras de projetos, fluxo de informação, procedimentos organizacionais podem ser explorados sem perturbar a continuidade das operações do sistema real;
- Novos projetos de equipamentos, arranjos físicos, sistemas de transporte podem ser testados sem a necessidade de utilizar os recursos existentes ou fazer aquisições;
- Hipóteses de "como" e "por que" certos fenômenos ocorrem podem ser testados para verificar a viabilidade;
- O tempo pode ser comprimido ou expandido permitindo o aumento ou redução da velocidade do fenômeno de investigação;
- O estudo da simulação pode ajudar o entendimento de como o sistema opera em vez de como os indivíduos pensam que o sistema opera.

As desvantagens são numeradas em relação ao custo x benefício, algumas vezes, as dificuldades e os gastos para implementação da simulação superam os resultados obtidos. Hoje, a relação das desvantagens está sendo superada pelas novas tecnologias e ampliação da capacitação dos recursos humanos na simulação, o que era considerado tendência vem se concretizando conforme apresentado na Tabela 1, (PORTO&PALMA, 2000).

**Tabela 1.** Soluções para as dificuldades da simulação,  
PORTO&PALMA (2000).

Dificuldades	Soluções
O desenvolvimento de modelos requer conhecimentos específicos.	Atualmente pacotes têm sido desenvolvidos contendo modelos genéricos que precisam apenas dos dados entrada para sua operação.
O levantamento de dados pode consumir muito tempo e ser oneroso.	Software de simulações serão capazes de obter dados de sistemas já existentes (integração)
Pode ser difícil interpretar os resultados de simulação	Softwares têm sido desenvolvidos com a capacidade de analisar os resultados da simulação.
A análise e a modelagem da simulação podem consumir muito tempo e ser dispendiosa financeiramente.	A modelagem é facilitada com a utilização de elementos de interface homem/computador e os avanços dos <i>hardwares</i> e <i>softwares</i> estão aumentando a velocidade da corrida da simulação.

Determinar o retorno de investimento em *software* de simulação de manufatura é uma tarefa difícil. A primeira razão é que o dinheiro salvo no final do processo é usualmente dinheiro que não foi planejado inicialmente. Simulação, quando aplicada antes de um sistema ser instalado, irá identificar problemas antecipadamente, prevenindo assim, custos com atrasos de lançamentos e repetição de atividades. Similarmente, quando a simulação é usada para estudar melhorias em sistemas existentes, irá evitar o gasto com alterações que não gerem as melhorias ou benefícios esperados, (PORTO&MIRANDA, 1998).

### **2.3 Simulação**

A simulação de sistemas é uma técnica de resolução de problemas seguindo as mudanças durante o tempo de um modelo dinâmico de um sistema. Um modelo de simulação é construído com mais liberdade do que aqueles desenvolvidos para uma solução delas analítica. Tipicamente, é construída uma série de seções (diagramas de blocos), cada uma descrita matematicamente sem uma preocupação excessiva com a complexidade. As equações precisam ser construídas e organizadas de modo que possibilitem o emprego de um procedimento para resolvê-las simultaneamente. (PORTO&MIRANDA, 1998)

A simulação possibilita a criação do modelo de um sistema real, com o propósito de avaliar o comportamento deste sistema sob várias condições. Permite ao analista tirar conclusões sobre novos sistemas sem precisar construí-los, ou fazer alterações em sistemas já existentes sem perturbá-los. Possibilita aos gerentes visualizar a operação de um sistema novo ou existente sob uma variedade de condições. É uma ferramenta que permite análise de interações entre sistemas (integração de sistemas) e entende como vários componentes interagem entre si e como estes afetam todo o desempenho do sistema.

A prática da simulação proporciona um total discernimento sobre a natureza de um processo para identificar problemas específicos ou áreas problemáticas dentro de um sistema. Além de desenvolver políticas ou planos específicos para um processo, testando novos conceitos e/ou sistemas antes de sua implementação.

Os resultados da simulação não serão precisos se os dados de entrada não forem precisos, e não descreverão características do sistema que não forem explicitamente modeladas.

### 2.3.1 Processo de Simulação

Tendo-se definido que, para análise de um dado sistema, um estudo de simulação é o que melhor se aplica; deve-se seguir certos passos para o estudo de simulação ser bem sucedido. Estes passos ou processo são conhecidos na literatura como “metodologia de simulação” ou “ciclo de vida de um modelo de simulação”, (CHWIF, 1999).

O processo de simulação consiste em definir, formular, validar e analisar uma sistema e recomendar as ações a serem tomadas. Fornece uma especificação funcional, um modelo de simulação e uma avaliação estatística para que os gerentes dos sistemas simulados tomem as iniciativas e decisões.

Uma especificação funcional é composta de objetivos, suposições, entradas e saídas. É detalhada no nível de lógica de controle, visando flexibilidade e análise.

### 2.3.2 Classificação da Simulação

Conforme NANCE (1993), a simulação pode ser classificada basicamente em três categorias: simulação Monte Carlo, simulação contínua e simulação de eventos discretos.

A simulação Monte Carlo é aquela onde um problema, notadamente não probabilístico, é solucionado por um processo estocástico. Nesta inexiste uma representação explícita do tempo, (CHWIF, 1999).

Simulação contínua e discreta são as duas maiores classes e levam em conta o tempo.

A simulação contínua trata das mudanças de estados com ocorrências contínuas e previamente conhecidas na linha do tempo. Nos sistemas contínuos, onde o interesse principal está em mudanças suaves, conjuntos de equações diferenciais são geralmente usadas para o cálculo das mudanças das variáveis de estado ao longo do tempo. Utilizada tipicamente para simular aproximações contínuas que inclui a modelagem do tempo (meteorologia), transformações de superfície e fluídos viscoso, (HYBINETTE, 2000); (PORTOMIRANDA, 1998).

---

Nos sistemas discretos, o interesse principal está nos eventos, as equações descrevem as condições para um evento ocorrer. A simulação consiste em efetivar as mudanças no estado do sistema, resultado da sucessão de eventos. Os sistemas de manufatura e ambiente virtuais são normalmente classificados como sistemas discretos.

Em alguns casos, é necessário construir um modelo de simulação que compreenda aspectos de eventos contínuos e discretos. Neste caso a simulação é denominada simulação combinada, conforme PRITSKER (1986), mas também é conhecida como simulação mista ou híbrida.

O trabalho apresentado na tese, concentra-se na simulação discreta, por esta razão este tópico é mais discutido.

### 2.3.2.1 Simulação de Eventos Discretos

Um Sistema de Eventos Discreto (SED) é aquele no qual o estado do sistema muda de acordo com um número finito de mudanças no tempo. Por exemplo, se um centro de usinagem produz uma peça acabada a cada três minutos, o número de peças acabadas produzidas pelo sistema (isto é, o estado do sistema), muda somente em instâncias discretas.

A simulação de eventos discretos começa pela definição matemática e lógica de como o estado do sistema irá mudar no tempo. Variáveis de estado são estabelecidas para coletar estatísticas sobre o desempenho do sistema. O tempo decorrido de um sistema é simulado por um relógio do sistema que dita o valor atual do tempo de simulação. Com o decorrer do tempo da simulação, ocorrem processamentos de acordo com um calendário de eventos que contém a lista de quando cada tipo de evento irá ocorrer no sistema. Os eventos podem ser as chegadas de material, início de operações, fim de operações, quebras de máquinas, conserto de máquinas etc. Um gerador de números aleatórios gera os valores de tempo de acordo com uma distribuição de probabilidades previamente especificada. Estes valores de tempo são usados para determinar a duração do evento. Por exemplo, o evento de chegada de peças pede por um outro evento que é o fim das operações. A diferença de tempo entre estes dois eventos é o tempo de operação. O tempo de operação pode ser exponencialmente distribuído, deste modo um gerador exponencial de números aleatórios gera um tempo de operação

---

exponencialmente distribuído, (CHWIF, 1999); (HYBINETTE, 2000); (PORTO&MIRANDA, 1998).

A simulação estudada neste trabalho é a de eventos discretos, portanto a partir deste ponto, quando não se mencionar o contrário, utiliza-se a palavra "Simulação", como sinônimo de "Simulação de Eventos Discretos".

### 2.3.3 Simulação Computacional

Embora a simulação computacional esteja datada da década de 50, quando tinha basicamente fins militares, sua popularidade intensificou-se nesta última década, expandindo-se para outras áreas como manufatura e serviços entre engenheiros e administradores e mesmos leigos. Esta popularidade pode ser explicada devido aos seguintes fatores, (CHWIF, 1999):

- Desenvolvimento dos computadores, que apresentou crescimento extraordinário nos últimos anos;
- Desenvolvimento de *softwares* de simulação com interfaces homem-máquina mais "amigáveis" e com capacidade de processamento gráfico;
- Natureza da simulação: capacidade de avaliar sistemas complexos e modelar seu comportamento dinâmico, sendo especialmente importante quando não existe uma solução analítica (através de modelos matemáticos).

Ainda, segundo PIDD (1992), pode-se enumerar algumas vantagens da simulação computacional quando comparada como a experimentação direta de um sistema real. A primeira é em relação aos custos envolvidos: é muito menos oneroso efetuar experiências com o modelo de simulação do que com o sistema real, especialmente quando há equipamentos de alto custo. Também a experimentação de sistema real envolve riscos, tanto materiais quanto humanos, o que não ocorre com um sistema simulado.

A execução da simulação normalmente é feita por um programa computacional. A simulação computacional contém (1) uma coleção de variáveis para designar o estado do sistema em questão; (2) um programa responsável em gerar mudança de estado nas variáveis do modelo com o passar do tempo; e (3) um protocolo de gerenciamento do tempo para administrar a execução do

---

programa. O tempo de simulação é a abstração do tempo real do sistema que está sendo modelado. O tempo de simulação define a ordenação dos eventos na simulação, (HYBINETTE, 2000) .

A simulação de eventos discretos normalmente mantém uma estrutura de dados do estado das variáveis, uma lista dos eventos que poderão ser ativados e um relógio global que indica o progresso do processamento computacional. O processamento de um evento pode causar a mudança de estado de uma ou mais variáveis e/ou um novo evento.

A modelagem do sistema a ser simulado pode ser implementada de duas maneiras diferentes: escrevendo-a em uma linguagem de propósito geral ou em uma linguagem especial de simulação. Muitas linguagens de simulação fornecem uma boa capacidade de modelagem do sistema. Todavia, estas linguagens têm limitações na capacidade de modelagem, muitas vezes por aparecerem na forma de blocos de funções. Para modelos mais sofisticados, os usuários devem escrever o programa usando linguagens de propósito geral. Várias linguagens de simulação também fornecem aos usuários uma série de funções úteis, tais como as para o gerenciamento dos eventos, para coletar e mostrar estatísticas, gerenciamento de filas, geração de números aleatórios etc. (HYBINETE, 2000); (CHWIF, 1999).

## **2.4 Modelagem**

O modelo é o corpo de informações sobre o sistema, ou seja, é a representação do sistema real que é utilizada para a realização do estudo de simulação.

O modelo deve conter todos os elementos que compõe o sistema real que são relevantes para as informações que se deseja obter, devendo assim, representar de maneira fidedigna o sistema estudado, de forma a garantir a qualidade das informações obtidas pela simulação.

O projetista além de colher as informações relevantes para o estudo que se deseja realizar deve, ao construir o modelo, levar em consideração qual o nível de abstração e quais as ferramentas de modelagem adequadas para o propósito em questão.

Os propósitos do uso de modelos são muitos, dentre eles, PORTO&MIRANDA (1998):

- permite ao investigador organizar suas crenças teóricas, suas observações empíricas e deduzir as implicações lógicas desta organização.
- permite uma maior compreensão do sistema;
- mostra as características mais relevantes;
- aumenta a velocidade em que a análise pode ser feita;
- permite testar as modificações do sistema;
- é mais fácil de manipular;
- permite controle de mais fontes de variação;
- é geralmente menos custoso.

O processo de determinação do grau com o qual o modelo corresponde ao sistema real ou ao documento de especificação é denominado "validação" do modelo. Proporcionar absoluta validade em relação ao sistema real é uma meta intangível. Por esta razão, o que realmente procura-se estabelecer é um alto grau no aspecto de validade. Por aspecto de validade se entende como sendo todas indicações externas que o levam a parecer com o sistema real ou especificado. Deste ponto de vista, a validação de um modelo é o processo de comprovar que este, dentro de seu domínio, é suficientemente preciso para a pretendida aplicação, (CHWIF, 1999). Portanto, o termo "validação" adotado neste trabalho tem este ponto de vista.

Não há testes simples para estabelecer a validade de um modelo. Validação é um processo indutivo, no qual o modelador extrai conclusões acerca da sua exatidão baseado nas evidências viáveis. Reunir evidências para determinar a validade do sistema está largamente associado ao exame da estrutura do modelo.

Os modelos podem ser classificados de vários modos. Há os modelos físicos, como o modelo de um avião, ou mais genericamente, uma réplica em escala reduzida. Há os modelos esquemáticos que incluem diagramas, mapas e plantas. Há os modelos simbólicos e os modelos matemáticos (equacionados).

---

Alguns são estáticos, outros são dinâmicos. Um modelo estático omite o tempo ou descreve o estado do sistema em um dado momento. Por outro lado os dinâmicos reconhecem explicitamente a passagem de tempo.

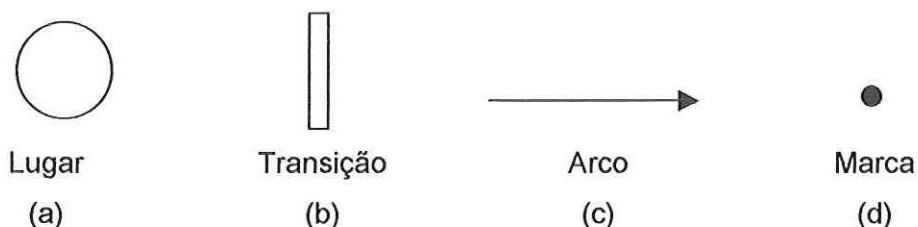
Uma outra distinção se refere aos modelos determinísticos e estocásticos. Num modelo determinístico entidades mantém relações matemáticas entre si. Como consequência essas relações determinam completamente as soluções. Num modelo estocástico, pelo menos parte da variação é de natureza aleatória. Assim podemos, no máximo, obter soluções médias pelo uso desses modelos.

A modelagem de um sistema de manufatura é feita de forma a levar em consideração todos os aspectos relevantes do mesmo. Existem várias técnicas para se fazer a modelagem, sendo que uma das principais é chamada de Redes de Petri, que pode representar Sistemas a Eventos Discretos graficamente ou matematicamente, (MURATA, 1989).

#### 2.4.1 Redes de Petri

A Rede de Petri (RP) foi introduzida em 1926 por Carl Adam Petri, e consiste numa ferramenta gráfica e matemática extremamente efetiva para a modelagem e análise de Sistemas de Eventos Discretos (SEDs), (KATO, 1999); (MURATA, 1989).

A rede de Petri consiste num grafo direcionado, com peso e bipartido, composto por dois elementos estruturais: lugares e transições. O lugar é representado graficamente por um círculo (figura 3a) e a transição por uma barra (figura 3b). Os elementos estruturais são utilizados para criar o modelo, no qual arcos orientados (figura 3c) conectam lugares a transições e transições a lugares. Estes arcos podem ser rotulados com um valor inteiro positivo, indicando o peso do arco. Um arco de peso  $k$  pode ser interpretado como  $k$  arcos paralelos de peso unitário.



**Figura 3.** Elementos da Rede de Petri

Na modelagem de Redes de Petri, utiliza-se o conceito de condições e eventos, lugares representam condições, e transições representam eventos. A presença de uma marca (Figura 3(d)) em um lugar é interpretado como condição verdade associada ao lugar. Uma outra interpretação, k marcas são colocadas em um lugar indicando que k dados ou recursos estão disponíveis. Uma transição (evento) tem um certo número de lugares de entrada e saída que representam as pré-condições e pós-condições de eventos, respectivamente. Os arcos fazem a ligação dos lugares de entrada/pré-condição com as transições e das transições com os lugares de saída/pós-condição.

Algumas interpretações de transições, lugares de entrada e lugares de saídas são mostradas na Tabela 2.

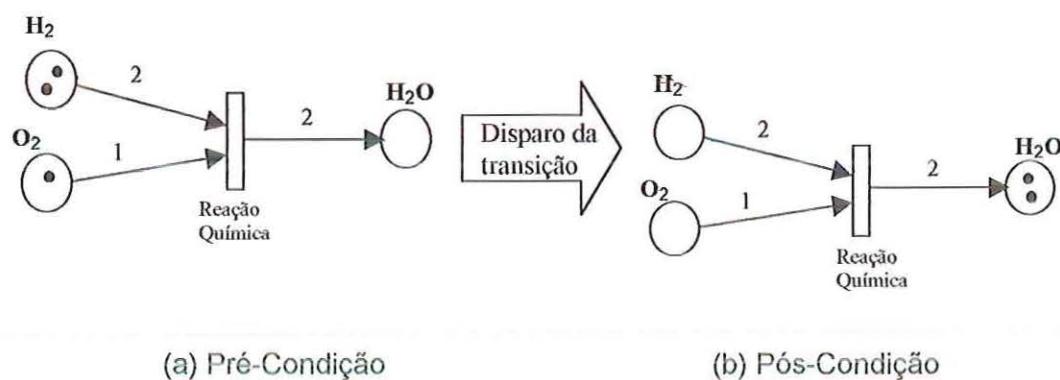
**Tabela 2.** Interpretações típicas de Transições e Lugares, (KATO, 1999).

Lugares de Entrada Pré-condição	Transições Evento	Lugares de Saída Pós-condição
Dado de entrada	Passo computacional	Dado de saída
Sinal de entrada	Processador de Sinal	Sinal de Saída
Recursos necessários	Tarefa ou trabalho	Recurso liberado
Condições	Cláusulas lógicas	Conclusões
<i>Buffers</i>	Processador	<i>Buffers</i>

Estes itens podem ser relacionados para modelar um problema. Por exemplo, a fórmula química  $2\text{H}_2 + \text{O}_2 \rightarrow 2\text{H}_2\text{O}$  possui a interpretação da tabela 3 e possui o modelo gráfico em RP da Figura 4.

**Tabela 3.** Interpretação da fórmula química ( $2\text{H}_2 + \text{O}_2 \rightarrow 2\text{H}_2\text{O}$ )

Lugares de Entrada Pré-condições	Transições Evento	Lugar de Saída Pós-condição
2 (duas) moléculas de $\text{H}_2$ e 1 (uma) molécula de $\text{O}_2$	Reação Química	2 (duas) moléculas de $\text{H}_2\text{O}$



**Figura 4.** Fórmula química ( $2\text{H}_2 + \text{O}_2 \rightarrow 2\text{H}_2\text{O}$ ) modelada em RP, (MURATA, 1989).

Os números nos arcos representam os pesos para que a reação química, simbolizada por uma transição, ocorra. Assim, dois átomos de hidrogênio mais um átomo de oxigênio formam uma molécula de água. Para sabermos se a reação ocorre é necessário que se tenha pelo menos duas marcas no lugar da molécula de hidrogênio ( $\text{H}_2$ ) (ou seja, duas moléculas  $\text{H}_2$ ) e uma marca na molécula oxigênio ( $\text{O}_2$ ) (ou seja, uma molécula de  $\text{O}_2$ ). Se isso ocorrer, as marcas serão tiradas dos lugares de entrada e serão colocadas duas marcas no lugar  $\text{H}_2\text{O}$ . Ou seja, a idéia geral é que os pesos dos arcos determinam quantas marcas deve haver em cada lugar de origem para que a transição seja disparada e seja colocada a quantidade de marca referente ao peso do arco de saída no lugar final.

Além de permitir a modelagem e representação gráfica de uma grande variedade de sistemas, a rede de Petri possui uma formalização matemática bem definida e técnicas de análise já consagradas, fundamentais para averiguação do

modelo. Dentre as técnicas existentes pode se destacar a verificação de propriedades comportamentais do modelo, determinação da árvore de alcançabilidade, uso da matriz de incidência e equação de estado, análise de invariantes (Apêndice A<sup>1</sup>), entre outras, (MURATA, 1989).

#### 2.4.1.1 Definição formal

A rede de Petri é uma quíntupla,  $PN = (P, T, F, W, M_0)$  onde, MURATA (1989):

$P = \{p_1, p_2, \dots, p_m\}$  é um conjunto finito de lugares,

$T = \{t_1, t_2, \dots, t_n\}$  é um conjunto finito de transições,

$F \subseteq (P \times T) \cup (T \times P)$  é um conjunto de arcos (relação de fluxo),

$W : F \rightarrow \{1, 2, 3, \dots\}$  é a função peso,

$M_0 : P \rightarrow \{0, 1, 2, 3, \dots\}$  é a marcação inicial (quantidade de marcas em cada lugar),

$P \cap T = \emptyset, P \cup T \neq \emptyset,$

A estrutura da rede de Petri  $N = (P, T, F, W)$  sem qualquer marcação inicial específica é denotada por  $N$ .

Uma rede de Petri com uma dada marcação inicial é denotada por  $(N, M_0)$ .

Além disto, denomina-se:

$M(p_i)$  é a marcação  $\in N$  em  $p_i$ ,

$\bullet p = \{t \mid (t, p) \in F\}$  é o conjunto de transições de entrada de  $p$ ,

$p\bullet = \{t \mid (p, t) \in F\}$  é o conjunto de transições de saída de  $p$ ,

---

<sup>1</sup> No Apêndice A é descrito as propriedades e métodos de análise de Redes de Petri levantadas por MURATA, (1989) em um artigo especial (Invited paper) para o Proceedings da IEEE (revisão de RP com mais de 300 referências).

$\bullet t = \{p \mid (p, t) \in F\}$  é o conjunto de lugares de entrada de  $t$ ,

$t\bullet = \{p \mid (t, p) \in F\}$  é o conjunto de lugares de saída de  $t$ ,

$R(N, M_0)$  ou simplesmente  $R(M_0)$  é o conjunto de todas possíveis marcações alcançáveis a partir de  $M_0$  e

$L(N, M_0)$  ou simplesmente  $L(M_0)$  é o conjunto de todas possíveis seqüências de disparos de transições a partir de  $M_0$ .

#### 2.4.1.2 Comportamento Dinâmico da RP

Cada lugar pode possuir marcas (marcas), indicando um estado. A marcação do sistema (estado), é denotada por um vetor  $M$  de dimensão igual ao número de lugares do modelo. O  $p$ -ésimo componente de  $M$ , indicado por  $M(p)$  consiste no número de marcas do lugar  $p$ . O vetor  $M_0$  consiste no vetor de marcação inicial, isto é, estado inicial do sistema.

O comportamento de diversos sistemas pode ser descrito em termos de seus estados e de sua respectiva mudança ou transição para outros possíveis estados. Na rede de Petri, a mudança de estado ocorre de acordo com a regra de habilitação e disparo das transições.

A ocorrência de um evento é denominada disparo de transição.

Uma transição está habilitada para disparar se para todo os lugares de entrada da transição, possuir um número de marcas maior ou igual ao peso do arco que conecta lugar a transições, no exemplo da Figura 4(a), a transição "reação química" está habilitada;

Uma transição habilitada pode disparar ou não, dependendo se o evento realmente ocorrer; no disparo de uma transição habilitada, todo lugar que possui um arco para a transição tem seu número de marcas reduzido pelo valor do peso deste arco, e todo lugar que possui um arco procedente da transição tem seu número de marcas acrescido do valor do peso deste arco. No exemplo da figura 4(b), lugares de entrada da transição foram decrementados e o lugar de saída da transição foi incrementado conforme os pesos do arco.

Uma transição sem lugares de entrada é denominada transição fonte e a sem lugares de saída é chamada de transição sumidouro. Uma transição fonte está sempre habilitada, e o disparo de uma transição sumidouro consome fichas, mas não produz nenhuma.

#### 2.4.1.3 Extensões de Rede de Petri

A rede de Petri básica possui algumas limitações tais, como a grande profusão de elementos gráficos em modelagem de sistemas complexos e a não inclusão do tempo de processo. Para contornar as limitações, existem muitas propostas de extensões. Nas aplicações mais abrangentes e complexas, torna-se vantajoso aproveitar conceitos e atributos de mais de uma extensão. [SOARES, 2001].

Algumas das extensões estão explicitadas a seguir, (CHWIF, 1999):

- **Redes de Petri Temporizada (TPN):** nesta extensão a duração de tempo  $\delta$  pode ser associado a transições ou lugares. No primeiro caso, se uma transição  $t$  dispara no tempo  $t$ , a marcação de entrada é removida e aparece na marcação de saída no tempo  $t+\delta$ . No segundo, se uma marcação aparece em  $P$  no instante  $t$ , então a transição posterior não pode ser disparada até o tempo  $t+\delta$ .
- **Redes de Petri Aumentadas (APN):** além da temporização, as marcações da rede podem ser criadas, destruídas, divididas ou condensadas. Ainda possuem outras características incorporadas, como arcos de controle
- **Redes de Petri Estocásticas (SPN/GSPN):** se uma Rede de Temporizada possui associados tempos exponencialmente distribuídos, temos uma Redes de Petri Estocástica. Se estas redes, por sua vez possuírem extensões das APN que generalizam as Redes de Petri, temos uma de Petri Estocástica Generalizada (GSPN).
- **Simulation Nets:** acrescenta arcos inibitórios, arcos de testes, propriedades temporais para transições, notação especial para representação de filas entre outras na RP básica.

#### 2.4.1.4 Softwares para Redes de Petri

A criação de softwares e ambientes para a edição e simulação de Redes de Petri teve a partir da década de 90 um grande impulso devido as evoluções tecnológicas na informática e ao fácil acesso destas tecnologias para os pesquisadores e usuários. Desta forma podemos destacar algumas contribuições importantes no que diz respeito a automação dos sistemas de manufatura (KATO, 1999).

Um depurador distribuído baseado em Redes de Petri que é projetado e implementado por LIU&ENGBERTS (1990), onde as principais funções suportadas são "breakpoints" distribuídos, execução passo a passo e repetição. O depurador consiste de um pré-processador o qual insere funções de controle dentro de um código fonte e gera um modelo em redes de Petri de um programa distribuído para monitoramento gráfico e simulação. O depurador também se comunica com programas existentes fornecendo acesso as suas variáveis. Também a superposição do depurador distribuído no topo de um programa seqüencial torna possível desacoplar programas seqüenciais em programas distribuídos.

Um modelo chamando PM-Net para representação e monitoração do processo de desenvolvimento de projetos é apresentado por LEE et al. (1994). Este modelo fornece informações para o progresso do gerenciamento em diferentes níveis de detalhes, para o auxílio de gerentes de projeto. O modelo enfatiza coleções de dados "bottom-up" e indaga funções de informação "top-down". O diagrama de fluxo de dados (DFD) e a técnica de *Work Breakdown Structure* (WBS) são usados para a construção de uma estrutura hierárquica de desenvolvimento do processo de software. Este processo pode ser visto como uma série de atividades. Cada atividade pode ser vista como uma série de sub-atividades, e cada sub-atividade pode ser vista como uma série de tarefas.

O ALPHA/Sim é uma ferramenta de simulação de eventos discretos de propósitos gerais (MOORE&BRENNAN, 1995). O ALPHA/Sim permite ao usuário construir graficamente o modelo de simulação, entrar com dados de entrada pelas vias integradas, executar o modelo de simulação e visualizar os resultados da simulação, dentro de um ambiente gráfico.

Uma ferramenta gráfica para modelagem em redes de Petri temporizadas é apresentada por LIU&ROBBI (1995), onde o objetivo foi o de desenvolver uma ferramenta computacional para desenho, edição e simulação de Redes de Petri temporizada usando programação orientada a objeto (POO). O desenvolvimento de uma ferramenta de simulação de Redes de Petri temporizada baseado em C++, chamada TiPNet, simula eventos discretos com transições estocásticas e imediatas. A depuração e o desempenho dos resultados são analisados nela.

O TimeNET da GERMAN, et al. (1995) é um pacote de software para modelagem e avaliação de Redes de Petri estocásticas (SPNs) na qual o disparo de tempo das transições pode ser exponencialmente distribuído, determinístico ou mais geralmente distribuído. Os modelos são especificados por de uma interface gráfica. Uma certa classe de distribuição de disparo no tempo pode ser usada, chamada de distribuição polinomial a qual é parte da especificação de polinômios exponenciais. Outros componentes especializados são fornecidos para a avaliação do modelo: análise estacionária, análise transitória, aproximação e simulação de componentes.

A ferramenta Petri Net Tools (2001) foi desenvolvida pelo grupo de pesquisa do laboratório de Simulação do Departamento de Engenharia Mecânica da Escola de Engenharia de São Carlos USP, consiste num editor e simulador gráfico de redes de Petri Lugar/Transição, desenvolvido com a metodologia orientada a objetos. Esta ferramenta tem uma estrutura ampliada incorporando características de um software aberto, o qual viabiliza o processo contínuo de desenvolvimento promovendo a inclusão, reutilização e aperfeiçoamento de funcionalidades. Busca-se assim incorporar a edição de outros tipos de redes de Petri, principalmente as interpretadas, pela reutilização de funcionalidades matemáticas e de edição gráfica da rede de Petri Lugar/Transição. As ferramentas de análise, disponíveis no módulo básico são: geração da matriz de incidência, geração de árvore de alcançabilidade, análise de propriedades comportamentais (alcançabilidade, limitabilidade, e vivacidade), análise usando invariantes (S-invariante e T-invariante), (SOARES, 2001).

## 2.5 Sistemas de Manufatura Discretas

A Modelagem e a Simulação neste trabalho estarão concentradas em Sistemas de Manufatura. Estes sistemas possuem características de sistemas de eventos discretos e são os responsáveis pela produção de uma empresa e incorporando tecnologias de gerenciamento e controle de todos os níveis produtivos

Três sistemas funcionais descrevem um sistema de manufatura (Figura 5), sendo que a união destes pode gerar um sistema com características de atender necessidades de flexibilidade e automatização e são: o sistema de transformação, o sistema de manuseio de materiais e o sistema de controle, (BEL et al., 1988) e (POLITANO, 1993).



Figura 5. Composição dos Sistemas de Manufatura, (KATO, 1999).

KATO, 1999 descreveu cada sistema funcional como:

- **Sistemas de Transformação** - são compostos por equipamentos que realizarão e/ou analisarão as transformações físicas na matéria prima para se atingir o produto final desejado. Fazem parte destes sistemas: - Máquinas ferramentas, Robôs industriais, Estações de montagem, Equipamentos de inspeção e controle de qualidade etc.
- **Sistemas de manuseio de materiais** - realizam o transporte e o armazenamento de materiais (peças, "pallets", ferramentas e fixadores), possuindo a função de interligar estações de carga/descarga, manufatura, inspeção e os armazéns. Os equipamentos que fazem parte deste sistema são: transporte (esteiras, empilhadeiras e transportadores manuais, AGVs (Veículos Auto Guiado), robôs etc.) e armazenamento (armazéns convencionais diversos, sistemas de armazenamento e

recuperação automático - AS/RS - "Automatic Storage/Retrieval Systems", *buffers*, etc.). Estes equipamentos fornecem ao sistema uma seletividade de carga, acesso aleatório às posições de armazenamento e flexibilidade de rota.

- **Sistemas de controle** - sistema de controle realiza o controle de fluxo de cada equipamento, coordena os elementos em trânsito, administrando o escalonamento das peças, operações e processos, armazenando dados internos/externos que auxiliem a tomada de decisões.

O sistema de controle é o responsável em integrar o Sistema de Manufatura, com o Sistema de Manuseio de Materiais e o meio externo (utilizando tecnologias como por exemplo CIM "Computer Integrated Manufactury"), além de controlar todos os evento, relativos às células.

### 2.5.1 Aplicações de Redes de Petri na Manufatura

Dentre as diversas aplicações, podemos citar a utilização para modelar componentes de Sistema Flexível de Manufatura (FMS) CECIL et al. (1992) e processos de fabricação PORTO (1990). Através dela é possível representar graficamente os estados dinâmicos (estados que podem ser assumidos durante o processo em questão) dos elementos de um sistema e seus eventos relacionados, como intertravamento de máquinas, coordenação de máquinas (robô, controle de comando numérico e Veículo Auto-Guiado), sequenciamento da produção, sincronismo de operações de montagem etc.

### 2.5.2 Visualização da Simulação de Sistemas de Manufatura

O objetivo principal do estudo da simulação é melhorar a qualidade das decisões administrativas, tornando desejável a animação gráfica, especialmente para modelagem de processos de manufatura. A animação permite um excelente meio de estabelecer a credibilidade para o modelo simulado. A realidade virtual é apontada como a principal ferramenta gráfica visual a ser utilizada, pois além da animação ela pode disponibilizar a imersão e a iteração, ou seja, o usuário teria sensação de estar dentro do modelo e poderia atuar no ambiente simulado em tempo real, permitindo uma exploração mais rica do mesmo.

Na execução dos experimentos e na análise dos resultados a RV é uma interface que pode fornecer um suporte importante para a tomada de decisões, pois a RV auxilia na criação de mundos virtuais semelhantes ao mundo real que ajudam na compreensão do funcionamento dos sistemas (PORTO&PALMA, 2000).

## 2.6 Realidade Virtual

Atualmente, um dos principais ápices da tecnologia é a realidade virtual. Porém, só recentemente ela se tornou disponível com plataformas a preços acessíveis para a maioria dos engenheiros de simulação. Agrupando-se algumas definições de realidade virtual, pode-se dizer que ela é uma técnica avançada de interface, onde o usuário pode realizar imersão, navegação e iteração em um ambiente sintético tridimensional gerado por computador, utilizando canais multisensoriais, (KIRNER&PINHO, 1996). A Figura 6 sintetiza o conceito de RV.

Na essência, a realidade virtual é um "espelho" da realidade física, ou seja, o indivíduo existe em três dimensões, tem a sensação do tempo e a capacidade de iterar com o mundo ao seu redor e a RV simula estas condições, chegando ao ponto em que o usuário pode "tocar" em objetos num mundo virtual e fazer com que eles respondam, ou mudem, de acordo com suas ações, (VON SCHWEBER&VON SCHWEBER, 1995).

A interface em realidade virtual envolve um controle tridimensional altamente iterativo de processos computacionais. O usuário entra no espaço virtual das aplicações, visualiza, manipula e explora os dados da aplicação em tempo real, usando seus sentidos, particularmente os movimentos naturais do corpo. A grande vantagem desse tipo de interface é que o conhecimento intuitivo do usuário a respeito do mundo físico pode ser transferido para manipular o mundo virtual. Para suportar esse tipo de iteração, o usuário utiliza dispositivos não convencionais como capacetes de visualização e controle, e luvas de dados, os chamados *data-gloves* EARNshaw, et al. (1995).

Os dispositivos fornecem ao usuário a impressão de que a aplicação está funcionando no ambiente tridimensional real, permitindo a exploração do ambiente e a manipulação natural dos objetos com o uso das mãos, por exemplo, para apontar, pegar, e realizar outras ações (KIRNER, 1996).



**Figura 6.** Realidade Virtual

A realidade virtual é freqüentemente confundida com animação, CAD ou multimídia, mas existem algumas características diferentes entre estes termos. Comparando a RV com estas tecnologias alternativas (LESTON, 1996), tem-se que a RV:

- é Orientada ao usuário;
- faz *rendering* (atualização da textura e da geometria do objeto) das imagens em tempo real, isto é, atualiza as imagens assim que sofrem qualquer tipo de modificação;
- é mais imersiva, pois existe a sensação de presença dentro do mundo virtual;
- é mais iterativa, pois o usuário pode influenciar no comportamento dos objetos e no estado do sistema;
- é mais intuitiva, há pouca ou nenhuma dificuldade em manipular as interfaces computacionais existentes entre o usuário e a máquina.
- é uma descrição topológica funcional dos objetos, ao contrário da descrição elementar do CAD.

Um sistema de RV envolve estudos e recursos ligados com percepção, hardware, software, interface do usuário, fatores humanos e aplicações BISHOP et al. (1992). Para a elaboração de sistemas de realidade virtual é necessário ter algum domínio sobre: dispositivos não convencionais de E/S, computadores de alto desempenho e boa capacidade gráfica, sistemas paralelos e distribuídos, modelagem geométrica tridimensional, simulação em tempo real, navegação, detecção de colisão, avaliação, impacto social, projeto de interfaces, e aplicações simples e distribuídas em diversas áreas (KIRNER, 1996).

### 2.6.1 Dispositivos de Realidade Virtual

Existem inúmeros documentos que falam sobre os dispositivos de RV e como eles atuam nos Ambientes Virtuais, (ENCARANAÇÃO, 1998) e (KIRNER, 1999). MACHADO (1995) fez uma síntese dos hardwares de RV.

- **Dispositivos Visuais** - Uma grande porção do cérebro é dedicada ao processamento e organização dos estímulos visuais. Devido a isto, os dispositivos visuais e o tipo de imagem gerada por um sistema de RV influenciam na determinação do nível de imersão de um sistema de RV.

Os sistemas de RV podem ser monoscópicos ou estereoscópicos, ou seja, cada um dos olhos pode visualizar ou não imagens diferentes. No caso de um sistema monoscópico, a mesma imagem será exibida para os dois olhos: apenas uma imagem passa pelo processo de rendering e é exibida para os dois olhos. Já no sistema estereoscópico, cada olho verá uma imagem ligeiramente diferente, sendo necessário um processo de rendering separado para cada uma das imagens.

Um outro fator importante quanto à parte visual da RV refere-se ao número de quadros por segundo que aparecem no vídeo, ou seja, a velocidade da simulação. Filmes projetados para o cinema apresentam aproximadamente 24 quadros por segundo, enquanto os projetados para TV apresentam aproximadamente 30 quadros por segundo. Em RV, buscam-se entre 15 e 22 quadros por segundo.

Existem duas classes de dispositivos visuais, a primeira é composta pelos videocapacetes (HMDs) e *head-coupled displays* (dispositivos que

utilizam-se de braços mecânicos para permanecerem diante do usuário), e a segunda é composta pelos monitores de computador e sistemas de projeção. O que diferencia estas duas classes é que na primeira existem sensores para os movimentos do usuário ligados ao dispositivo visual, enquanto que na segunda isso não ocorre e tudo vai depender dos comandos do usuário via outro dispositivo de entrada

- **Dispositivos Auditivos** - Os dois ouvidos captam ondas sonoras provenientes de todas as direções. O formato de concha do ouvido externo capacita-o para o trabalho de coletar ondas sonoras e direcioná-las para os vários caminhos através do canal auditivo. O cérebro então recebe e processa as características deste som para determinar ou localizar o local exato da fonte sonora. Os sistemas de som 3D duplicam artificialmente os ativadores naturais que auxiliam o cérebro a localizar o som, além de recriar eletronicamente esses efeitos em tempo-real.

Existem diversas placas de som projetadas para trabalhar com conjuntos de ferramentas que constróem mundos em RV. Algumas dessas placas permitem trabalhar com diversas fontes de som simultâneas. O método mais popular para criar e controlar sons é o MIDI (musical instrument digital interface).

- **Dispositivos Físicos** - Os dispositivos físicos procuram estimular as sensações físicas, como o tato, tensão muscular e temperatura. Diferente dos dispositivos de saída de visão e audição, os dispositivos físicos requerem uma sofisticada iteração eletromecânica com o corpo do usuário. A tecnologia existente atualmente não é capaz de estimular os sentidos físicos com o nível de realismo que atinge os sentidos visuais e auditivos: o problema está além da criação de dispositivos de feedback, pois envolve também a compreensão e simulação das forças apropriadas.
- **Dispositivos de Iteração** Existem diferentes dispositivos de iteração com diferentes finalidades: é importante escolher o mais adequado para a aplicação de RV em questão. A escolha do dispositivo de iteração mais adequado leva em conta não apenas a finalidade do sistema, mas

também o software utilizado, pois a eficiência do sistema vai depender da capacidade do software aproveitar as características do dispositivo.

Existe uma série de dispositivos de iteração disponíveis atualmente, variando desde luvas de dados até dispositivos chamados de sensores biológicos

- **Dispositivos de Trajetória** - Muitos dos dispositivos de iteração mencionados acima contam com um dispositivo responsável pela tarefa de detecção ou rastreamento da trajetória, conhecido como dispositivo de trajetória ou tracking.

Os dispositivos de trajetória trabalham baseados na diferença de posição ou orientação em relação a um ponto ou estado de referência. Basicamente existe uma fonte que emite o sinal (que pode estar localizada no dispositivo de iteração), um sensor que recebe este sinal e uma caixa controladora que processa o sinal e faz a comunicação com o computador.

### 2.6.2 Conceitos de Imersão, Iteração e Envolvimento

A realidade virtual também pode ser considerada como a junção de três idéias básicas: imersão, iteração e envolvimento, (MORIE, 1994). Isoladamente, essas idéias não são exclusivas de realidade virtual, mas aqui elas coexistem.

A idéia de imersão está ligada com o sentimento de estar dentro do ambiente. Normalmente, um sistema imersivo é obtido com o uso de capacete de visualização, mas existem também sistemas imersivos baseados em salas com projeções das visões nas paredes, teto e piso, (CRUZ-NEIRA et al., 1992). Além do fator visual, os dispositivos ligados com os outros sentidos também são importantes para o sentimento de imersão, como: som, posicionamento automático da pessoa e dos movimentos da cabeça, controles reativos etc. A visualização tridimensional pelo monitor é considerada não imersiva. Dessa forma tem-se a conceituação de realidade virtual imersiva e não imersiva, (LESTON, 1996).

Do ponto de vista da visualização, a realidade virtual imersiva é baseada no uso de capacete ou salas de projeção nas paredes, enquanto a realidade virtual

---

não imersiva baseia-se no uso de monitores. De qualquer maneira, os dispositivos baseados nos outros sentidos acabam dando algum grau de imersão à realidade virtual com o uso de monitores, mantendo sua caracterização e importância (ROBERTSON et al., 1993). Embora a realidade virtual com uso de capacetes tenha evoluído e seja típica, a realidade virtual com monitor apresenta ainda alguns pontos positivos como: utilizar plenamente todas as vantagens da evolução da indústria de computadores; evitar as limitações técnicas e problemas decorrentes do uso do capacete, baixo custo e facilidade de uso. Porém é importante frisar que a tendência será a utilização da realidade virtual imersiva para a grande maioria das aplicações futuras.

Com relação à idéia de iteração, ela está ligada com a capacidade do computador detectar as entradas do usuário e modificar instantaneamente o mundo virtual e as ações sobre ele (capacidade reativa). As pessoas gostam de ficar cativadas por uma boa simulação e ver as cenas mudarem em resposta aos seus comandos, esta é a característica mais marcante nos vídeo-games.

Para que um sistema de realidade virtual pareça mais realista, objetos simulados são inseridos no ambiente virtual. Outros artifícios para aumentar o realismo são empregados, como por exemplo, a texturização dos objetos do ambiente e a inserção de sons tanto ambientais quanto sons associados a objetos 3D específicos, (ARAÚJO, 1996).

A idéia de envolvimento, por sua vez, está ligada com o grau de motivação para o engajamento de uma pessoa com determinada atividade. O envolvimento pode ser passivo, como ler um livro ou assistir televisão, ou ativo, ao participar de um jogo com algum parceiro. A realidade virtual tem potencial para os dois tipos de envolvimento ao permitir a exploração de um ambiente virtual e ao propiciar a iteração do usuário com o mundo virtual dinâmico [KIRNER & PINHO,96].

### 2.6.3 RV Passiva, Exploratória ou Iterativa

Quanto à utilização do aplicativo de realidade virtual, ele pode fornecer uma sessão em três formas diferentes: Passiva, exploratória e iterativa (ADAMS, 1994).

Uma sessão de realidade virtual passiva proporciona ao usuário exploração automática e sem interferência através do ambiente 3D. A rota e as vistas são

---

explícitas e exclusivamente controladas pelo software. O usuário não tem controle algum, exceto talvez, para sair da sessão, (VALERIO, 1998).

Uma sessão de realidade virtual exploratória proporciona uma exploração dirigida pelo usuário através do ambiente 3D. O participante pode escolher a rota e as vistas, mas não pode iterar com entidades contidas na cena 3D.

Uma sessão de realidade virtual iterativa proporciona uma exploração dirigida pelo usuário através do ambiente 3D. Além disso, as entidades virtuais no ambiente 3D respondem e reagem às ações do participante. Por exemplo, se o usuário move o ponto de observação em direção à porta, a porta pode se abrir, permitindo ao participante passar através dela. Os sensores do gerenciador de simulação detectam quando o nó do ponto de observação se move dentro de um volume especificado do espaço-3D. O gerenciador de simulação então chama uma função para animar a abertura da porta no espaço cibernetico (*cyberspace*). O gerenciador de simulação em questão é um instrumento de animação modificado para rodar no modo passo a passo para uso em sessão de realidade virtual. Ele executa funções *run-time* como atualizar, impor, detectar e mover, isto é, ele atualiza a imagem na tela, impõe as regras de realidade virtual, detecta iterações e move entidades.

Este espaço cibernetico citado é definido como o espaço-tempo 4D simulado, controlado pela interface de realidade virtual. O espaço cibernetico existe somente dentro do computador, isto é, trata-se de um espaço imaginário. A realidade virtual, por outro lado, é a interface homem-computador que permite ao usuário iterar com o espaço cibernetico. Talvez possa ser mais conveniente se pensar no espaço cibernetico como sendo a imaginação do computador (ADAMS, 1994).

#### 2.6.3.1 Tipos de Realidade Virtual

Segundo KIRNER&PINHO (1996) existem quatro tipos básico de aplicações em Realidade Virtual, Sistemas de Telepresença, Sistemas de Realidade Virtual, Sistema de Realidade Aumentada e Sistema de Realidade Melhorada.

---

As aplicações mais comuns são Sistemas de Realidade Virtual, o usuário participa de um mundo virtual gerado no computador, usando dispositivos sensoriais de percepção e controle.

Num Sistema de Telepresença a pessoa está objetivamente presente num ambiente real e separada fisicamente no espaço. São sistemas onde um usuário controla um dispositivo remoto (como um telerobô) que está fisicamente posicionado em outro local. A intenção é fazer com que o usuário sinta-se como se estivesse no lugar do dispositivo.

Dois subconjuntos da área de realidade virtual utilizam tecnologias diferentes para proporcionar um aumento do desempenho humano, trata-se da realidade aumentada - RA (*Aumented Reality*) e realidade melhorada - RM (*Enhanced Reality*).

A RA combina imagens geradas no mundo virtual com imagens do mundo real por meio do uso de um capacete parcialmente transparente provido de sensores. O objetivo destes sistemas é suplementar a informação obtida no mundo real com informações geradas pelo computador. O objetivo destes sistemas é suplementar a informação obtida no mundo real com informações geradas pelo computador.

A RM explora a convergência das tecnologias de visão de máquina com computação gráfica, como uma forma intermediária entre o mundo real e o mundo da ficção.

Basicamente, a diferença entre RA e RM é que na primeira, a realidade é suplementada por ambientes sintetizados pelo computador enquanto que na segunda, a realidade é suplementada por ambientes gerados através de uma combinação de vídeo e computação gráfica, (ARAÚJO, 1996).

#### **2.6.4 Realidade Virtual na Manufatura**

Conforme o COMMITTEE ON VISIONARY MANUFACTURING CHALLENGES (1998) existem dois elementos cruciais e necessários para o sucesso modelagem e simulação dos sistemas de manufatura: o modelo e a

---

interface homem/maquina que habilitam os indivíduos a iterar com os modelos para aprender, planejar e fazer o controle da manufatura.

Os indivíduos são componentes críticos em qualquer sistema de manufatura. Modelos e simulações devem considerar os dois pontos de vistas dos indivíduos. Primeiro, o comportamento e ações dos indivíduos devem ser incluídas nos modelos, como parte de um sistema manufatura. Isto implica numa compreensão de como os indivíduos se relacionam uns com os outros dentro do sistema, e também como os indivíduos se relacionam com os equipamentos e processos (que podem ou pode não ser automatizado) segundo, modelos e simulações que devem ser descritas e entregues num formato utilizável para facilitar a decisão ou ação a serem tomadas, (*COMMITTEE ON VISIONARY MANUFACTURING CHALLENGES*, 1998).

As principais vantagens que a interface de RV oferece na área da manufatura são, (VALÉRIO, 1998):

- projetar máquinas que podem ter suas propriedades estruturais e funcionais avaliadas e testadas;
- desenvolver uma ergonomia funcional e confiável, sem ter que construir um modelo em escala real;
- projetar produtos que possuam *design* estético de acordo com a preferência de cada cliente;
- facilitar operações remotas e controle de equipamentos (telemanufatura e telerobótica);
- desenvolver e avaliar processos que assegurem a manufaturabilidade, sem ter que de fato produzir o produto em escala comercial;
- desenvolver planos de produção e itinerários, e simular se esses estão corretos e
- educar empregados em técnicas avançadas de manufatura, principalmente com ênfase em segurança no trabalho.

#### **2.6.4.1 Visão geral sobre o emprego da realidade virtual na indústria**

A cada instante surgem novas aplicações nas mais variadas áreas do conhecimento e de maneira diversificada, em função da demanda e da capacidade

---

criativa das pessoas. Em muitos casos, a RV vem revolucionando a forma de iteração das pessoas com sistemas complexos tratados com o uso de computadores, propiciando maior desempenho e diminuindo os custos (VALERIO,PORTO&PALMA, 1997).

O uso da RV para dar suporte a aplicações industriais estão rapidamente crescendo, (PENG, 2000) e (ROSEMBLUM & MACEDONIA, 1998).

Para LESTON (1996) as empresas tem utilizado a RV nos seguintes campos: automação de projetos, venda e *marketing*, planejamento e manutenção, treinamento e simulação, e concepção e visualização de dados.

O COMMITTEE ON ADVANCED ENGINEERING ENVIRONMENTS, AERONAUTICS AND SPACE ENGINEERING BOARD (2000) levantou que os investimentos em tecnologias de Ambientes Avançados de Engenharia - AEE (Advanced Engineering Environment ) a curto prazo fornecem um baixo retorno, mas a longo prazo seriam pagos. As empresas que começaram cedo a aplicar AEE, aprenderam muito através dos erros, desta forma, junto com as expectativas mais realísticas e , conduziram os maiores benefícios e satisfação a cada nova geração de tecnologias de AEE.

Alguns exemplos de aplicações pelas indústria e empresas de cunho tecnológico foram citados abaixo.

O Intelligent Synthesis Environment (ISE) da NASA (National Aeronautic and Space Administration) tem o propósito de desenvolver tecnologias de AEE e sistemas. Quase todas avaliações seriam feitas virtualmente, não fisicamente, com ambientes imersivos operados geograficamente e ritmicamente distribuídos as equipes colaboradoras. O ISE também tem que apoiar missões como as das sondas espaciais, onde não será práctico a supervisão humana. (COMMITTEE ON ADVANCED ENGINEERING ENVIRONMENTS, AERONAUTICS AND SPACE ENGINEERING BOARD, 2000)

A BMW usa a RV para explorar a visualização dos *layouts* dos novos carros para aumentar a comunicação e a simulação dos processos, enquanto a Rolls Royce Aeroengines And Associates usa RV para os procedimentos de manutenção. O treinamento de engenheiros de manutenção por meio destes

---

procedimentos aumenta a visualização de ambientes complexos e das prototipagem rápidas da planta de controle e instrumentação (PENG,2000).

Na área militar, um exemplo típico de utilização é a simulação de uma cabine de avião de combate, que foi desenvolvida pela British Aerospace Real para treinamento dos cadetes britânicos, (KALAWSKY, 1993). Outro trabalho na área de simulação e treinamento em aviões de combate é citado por MCCARTY et al. (1994). A realidade virtual também é usada para treinar operadores de radares que rastreiam trajetórias de aeronaves SENSE8 (2001) e na simulação de um tanque de guerra para treinamento, este trabalho é vinculado ao projeto SIMNET desenvolvido pelo DARPA (Defense Advanced Research Projects Agency - USA) que viabiliza um ambiente virtual distribuído onde vários simuladores virtuais remotos ficam interligados entre si, trocando informações e mantendo atualizados a descrição deste mundo, (PIMENTEL & BLAU, 1994); (MOSHELL et al., 1994) e (ELLIS, 1994).

A realidade virtual também está sendo empregada em projetos relacionados ao programa espacial de vários países, por exemplo a European Space Agency (ESA) utiliza esta tecnologia para projetar e desenvolver sistemas de simulação para treinamento e aprendizagem dos astronautas, em diversos equipamentos utilizados no projeto espacial BAGIANA (1993) e ENCARNAÇÃO et al. (1994). A NASA utilizou a RV para criação de um AV para treinamento do grupo encarregado de dar manutenção e fazer os reparos necessários no telescópio espacial Hubble LOFTIN&KENNEY (1995), e para o desenvolvimento de técnicas de programação de robôs através da simulação de ambientes de tarefa remotos (ELLIS, 1994). Mais recentemente, a RV foi utilizada pela NASA no projeto de exploração ao planeta Marte SENSE8 (1997).

A Haywood Community College (Waynesville, NC) utiliza RV para que seus estudantes tenham uma melhor visualização e iteração com seus modelos, feitos em 3D a partir do AutoCAD SENSE8 (1996). O laboratório de pesquisas da Nippon Electric Company (NEC) desenvolveu um sistema de RV para permitir que os operadores usem os movimentos de suas mãos (datagloves) para manipular modelos tridimensionais de CAD, (KAHANER, 1994).

A empresa Fujita (Tsukuba, Japão) está aplicando tecnologia de realidade virtual na construção e controle de robôs. A meta é desenvolver sistemas remotos de controle, que permitam a manipulação dos robôs por cursores ou ponteiros de exibição com interfaces gráficas.

### 2.6.5 Realidade Virtual e Simulação

A tecnologia da simulação teve um crescimento rápido desde os seus primórdios na década 1960. Agora, muitas áreas confiam no uso intensivo da simulação para testar novas idéias e opções.

Tradicionalmente, os passos seguidos pela simulação de manufatura à eventos discretos são: preparar variáveis de entradas, selecionar parâmetros, rodar a simulação e revisar os resultados após a execução. A principal desvantagem de tais sistemas é que eles são complexos para analisar “O que se ...” e requer vários ciclos antes de divulgar informações interessantes e valiosas. A nova abordagem é utilizar os conceitos computacionais tradicionais e juntar a iteratividade dos ambientes virtuais para guiar simulação da manufatura rapidamente, assim permite mudanças rápidas e retorno instantâneo do sistema para a obtenção de resultados (KESAVADAS&SUDHIR, 2000).

As indústrias mais ousadas estão extrapolando as aplicações tradicionais com o uso da Realidade Virtual e da simulação iterativa baseada em jogos futurísticos. O grau de envolvimento através da imersão do usuário dentro de um ambiente virtual está crescendo continuamente e uma grande variedade de aplicações de simulação já está disponível para os usuários na indústria de manufatura, (PORTO&PALMA, 2000).

A simulação se tornará o caminho para a realização dos negócios na fábrica futuro, pois as principais decisões sempre necessitarão ser avaliadas em todos os seus aspectos. O uso da modelagem de simulação se estenderá das aplicações tradicionais da manufatura e aspectos logísticos à processos de negócio e à aplicações de simulações iterativas no treinamento e vendas, penetrando em todos os aspectos das corporações e em muitos aspectos das vidas pessoais. (PORTO&PALMA, 2000).

LOBÃO & PORTO (1996) afirmam em seu artigo que a nova tendência para simuladores de eventos discretos, aponta para sistemas iterativos e inteligentes, nos quais serão largamente empregadas técnicas de realidade virtual, inteligência artificial e sistemas especialistas. A realidade virtual seria utilizada inicialmente no módulo de animação, que é a parte do software que mostra a iteração entre os diversos componentes do modelo, durante a execução de uma corrida de simulação. Dessa forma, permitirá ao usuário iterar com os componentes do sistema durante o seu "funcionamento virtual", bem como, imergir no interior do modelo dando mais realismo ao sistema e permitindo uma exploração mais rica do mesmo.

JONES&CYGNUS (1993) expõe como a realidade virtual pode ser agregada a softwares comerciais de simulação de manufatura. Concluindo que a implementação de uma interface em RV é viável comercialmente, fornecendo um suporte maior para o apoio à decisão e aprofundando o entendimento do modelo de simulação.

Na área de simulação de chão de fábrica está havendo um grande esforço de empresas de simulação e universidades para desenvolver um ambiente para estudos dos vários aspectos de um modelo imersivo de exibição baseado em técnicas de modelagem de realidade virtual. Como, por exemplo, o desenvolvimento de um modelo de fábrica de engrenagens da Figura 7 baseados em softwares avançados de RV, (VALÉRIO, PALMA & PORTO, 1997).

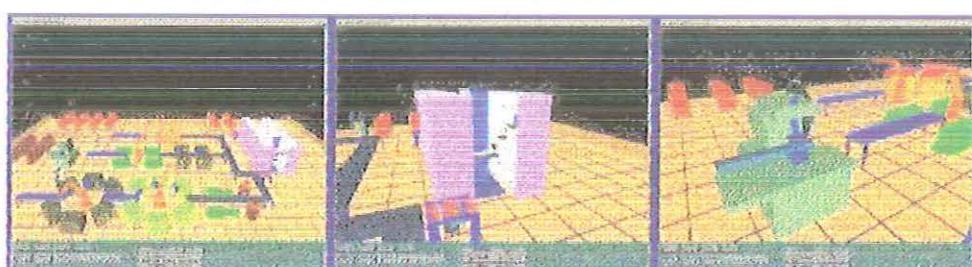


Figura 7. Vistas da fábrica de engrenagens virtual, BANERJEE et al. (1997).

JAIN (1999) comenta que as mudanças mais dramáticas na área de simulação serão na capacidade de visualização. Esta área vem evoluindo rapidamente, e isto já pode ser observado nos avanços das capacidades e das variações dos softwares comerciais de simulação de manufatura com visualização

3D à Realidade Virtual (RV) imersiva em institutos de pesquisas . O uso de ambiente de RV imersivas se tornará comum no futuro. O usuário será capaz de entrar dentro do modelo e experimentar o sistema de simulação completo e com os sons.

Há poucas ferramentas de edição de modelos para simulação de sistemas de manufatura com interface de RV, normalmente elas são direcionado a uma sub-área da manufatura, possuem elevado custo, e a tecnologia empregada é segredo industrial.

### **2.6.6 Ambientes especializados de realidade virtual**

Existem diversas empresas especializadas em softwares de ambientes virtuais, algumas possuem modeladores para desenvolvimento de aplicações gerais, DIVISION (2001); SENSE8 (2001); SUPERSCAPE INC. (2001); O'NEILL (1995) e outras para planejamento e implementação de mundos virtuais em tempo real. BLANCHARD et al. (1990); ENCARNAÇÃO&FRAUNHOFER. (1998) e outras se especializaram em desenvolvimento de ambiente virtual iterativo para projeto, análise e prototipação virtual.

Por exemplo, existem softwares específicos para ferramenta de simulação e programação para solda a arco e para solda a ponto, programação de pintura robotizada, novo padrão para simulação de eventos discretos, ferramenta de simulação e análise para centros de usinagem e programas NC, ferramenta para simulação e análise de ergonomia e iteração humana na engenharia e desenvolvimento iterativo de produtos e processos Deneb Robotics, Inc., é um subidisiaria Dassault Systèmes- DELMIA (2001).

Empresas como a Dassault Systemes (Paris, França) e DCT Inc. (Detroit, EUA) INTELLIGENT MANUFACTURING (1995) se especializaram em sistemas CATIA/CADAM da IBM para simular, validar e integrar células de manufatura robotizadas de soldagem e montagem, utilizadas em empresas automotivas.

Existem também softwares especializados na projeção, otimização e programação de células de trabalho virtuais nas quais operadores humanos trabalham junto com robôs, é o caso do ROBOCAD/Man, da Tecnomatrix, OWEN (1994b). Este sistema permite a execução de diversos tipos de operações, tais

---

como manuseio de materiais para fabricação e montagem. O sistema também é utilizado para determinar os melhores procedimentos de montagem do produto, uma vez determinada a rotina de fabricação, o sistema pode ser empregado no treinamento dos operadores na realização das tarefas a serem desempenhadas pelos mesmos. Este software gera ainda relatórios ergonômicos, permitindo a predição dos efeitos físicos sofridos por cada trabalhador na realização de tais tarefas, (DEITZ, 1995).

Na linha de softwares especialistas de realidade virtual, ainda podemos citar: o Flexman Simulation, o ADAMS/Car, VisMockUp e o VEDAM.

- O software Flexman Simulation desenvolvido pela Lockheed P.A. Research Laboratories (Palo Alto, CA) é utilizado para otimizar sistemas flexíveis de manufatura (*Flexible Manufacturing System - FMS*). Ele permite mapear operações paralelas, especificar quais iterações existirão periodicamente e posteriormente armazena informações do sistema numa base de dados temporal. Utilizando avançadas tecnologia computacionais, o software simula minuto a minuto as operações de um FMS realizadas sob diversas condições. O usuário tem acesso a modificação de algumas variáveis de manufatura como, por exemplo: *setup* do chão de fábrica, *setup* das ferramentas e definição e seleção de tarefas (OWEN, 1994a).
- O software ADAMS/Car permite que engenheiros criem modelos computacionais de veículos com exata representação de partes envolvidas na montagem como: suspensão, motor, mecanismos de direção, tração, freios e outros sistemas de controle. Os engenheiros são capazes de testar e validar o modelo virtual em diversas condições de estradas, para poderem avaliar precisamente algumas características como: qualidade da viagem, segurança do veículo e parâmetros de performance do *design*, (TERESKO, 1995).
- O VisMockUp desenvolvido pela empresa Engineering Animation Inc. (Ames, Iowa) permite aos engenheiros montar várias peças de um conjunto, por exemplo, a suspensão de um veículo e depois posicioná-las em situações extremas para fazer um estudo das possíveis

interferências. É permitido identificar os lugares que apresentaram problemas, e posteriormente assinalar estas áreas, para acrescentar comentários que irão auxiliar num futuro reprojeto do conjunto. O software de prototipação virtual também permite fazer medidas e modificações nos atributos das peças, (DVORAK, 1997).

- O VEDAM é um sistema de desenvolvimento de ambientes virtuais para design e manufatura (EXHIBITORS, 1997). O VEDAM utiliza parâmetros vindo de sistemas CAD/CAM, como os fornecidos pelo software Pro/Engineer. O software consiste de quatro ambientes virtuais: de modelamento de máquina, de projeto, de montagem e de manufatura. As interfaces do VEDAM com os sistemas CAD são feitos através de um integrador de dados que permite um fluxo bidirecional de dados entre ambos.

O modelamento de máquina é um ambiente não imersivo utilizado para criação de máquinas-ferramenta encontradas no chão de fábrica, como por exemplo: tornos, fresadoras e furadeiras. Não somente a geometria pode ser criada como também o funcionamento das máquinas podem ser simulados. O projeto virtual é utilizado para auxiliar na concepção de partes tridimensionais do ambiente. No ambiente de montagem virtual é feita a montagem das partes referentes ao ambiente que está sendo desenvolvido e no ambiente de manufatura virtual são testados programas de comando numérico nas máquinas pré-definidas pelo sistema (ANGSTER&JAYARAM, 1997).

## 2.7 Modelagem e Análise de AVs com RP

Conforme MURATA (1989) a RP possui características como: modelagem através de vários níveis de abstração; modelagem de sistemas paralelos e concorrentes; sincronização de eventos; verificação de propriedades de forma sistemática.

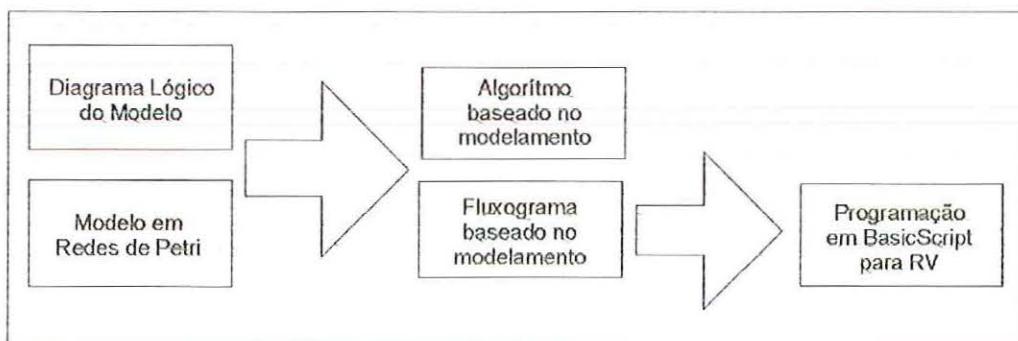
E conforme SMITH&DUKIE, 1999 a avaliação formal do sistema RV é desejada, mas é comum as descrições de AVs utilizarem um nível informal. Pois, AVs são ambientes dinâmicos e possui natureza altamente contínua e visual,

extremamente difícil definir os aspectos salientes e úteis. AVs são constituídos de uma coleção complexa de processos discretos e contínuos.

A implementação e o teste de ambientes virtuais são complicados devido às concorrências e as características de tempo real. Portanto, o desenvolvimento de métodos formais para a modelagem e análise é altamente desejado, e eles devem modelar a sincronização dos processos assíncronos, que é muito comum em sistemas de RV.

Conseqüentemente a RP é uma ferramenta bem apropriada para a modelagem de ambientes virtuais, pois pode capturar naturalmente as principais características de sistemas de RV, e proporciona resultados empíricos muito bons na verificação automática de concorrência e sistemas de tempo real por permitir modelar facilmente a computação não determinística e paralela (MASCARENHAS, 1998) e (ZHOU-YI, et al., 2000a).

Algumas implementações de AV utilizam a RP para fazer a modelagem do comportamento do ambiente e as análises, depois transformam a RP em várias pequenas rotinas de programação imperativa (seqüencial) e insere em ferramentas de RV como *script* de uma determinada atividade do AV, como no trabalho do VALÉRIO, 1998, conforme Figura 8.



**Figura 8.** Esquema de obtenção das rotinas para o ambiente virtual (VALÉRIO, 1998)

Há vários trabalhos que aplicam modelo de RP para a modelagem de ambientes virtuais, por exemplo, o controle do comportamento: dos modelos (formas) gráficos (FISHWICK, 2000), do treinamento de manutenção em máquinas virtuais (ISHII, et al., 1998) e (YOSHIKAWA, et al., 1997), dos equipamentos virtuais

como: robôs (HALE, et al., 1999) e (MO&TANG, 1998), submarinos (BRUZZONE, et. al., 1999), CNC para treinamento (LIN, et al. 1999) e (VALÉRIO, 1998).

MASCARENHAS (1998), utilizou RP para fazer a averiguação de um sistema computacional pronto, ele fez a modelagem e análise em RP de um CAVE (sistema de baseado na projeção nas paredes, chão e teto de imagens de um ambiente sintético geradas pelo computador).

ZHOU-YI, et al. (2000a) e ZHOU-YI, et al. (2000b) desenvolveram um trabalho para a modelagem de um ambientes virtuais em rede (NICE - *Narrative Immersive Constructionist / Collaborative Environment*). Eles utilizaram uma extensão de Redes de Petri a RP fuzzí-temporizada para a modelagem e simulação de ambientes de realidade virtual em rede.

A necessidade crescente da utilização de recursos de modelagem para projetar ambientes virtuais gerou recentemente alguns ambientes baseados em RP, como pode ser visto nos trabalhos de ECONOMOU (2001); SHIMIT, et al. (1999); WILLANS, et al. (2001); ZHANG, et al. (1999) e ZHANG, et al.(2000). SHIMIT&DUKIE (1999), propõe uma ferramenta para construção de sistemas controles genéricos para ambientes virtuais híbridos, que mistura a rede de petri junto com outros elementos de sistemas contínuos.

## 2.8 Engenharia de Software - Orientação a Objetos

No campo da engenharia de software, a metodologia de desenvolvimento de sistemas orientados a objeto tem se mostrado uma poderosa ferramenta para implementação de sistemas complexos devido às seguintes características (KHOSHAFIAN, 1990).

- Modelagem do mundo real mais próxima à ótica do usuário;
- Facilidade de iteração com ambiente computacional, usando metáforas familiares;
- Construção de componentes reutilizáveis de software e facilidade de expandir bibliotecas nos módulos do software;
- Facilidade de implementar modificações e extensões em componentes sem a necessidade de uma exaustiva re-codificação.

A representação do sistema através de objetos, por ser mais próxima à ótica humana, permite uma maior facilidade de visualização do funcionamento e do fluxo de informação deste. A implementação através deste paradigma fica extremamente mais simples e apurada em relação a usual técnica de programação estruturada. (SOARES, 2001).

A rede de Petri, por ser uma técnica de modelagem formal e bem definida, possui atributos gráficos e matemáticos que beneficiam uma implementação orientada a objeto, (SOARES, 2001).

Para desenvolver sistemas computacionais com interface de RV, o paradigma de Orientação a Objetos é muito empregado, pois, a RV representa o real em forma digital, e o paradigma O.O. propõe fazer a abstração do problema utilizando ferramentas para a computação com um formato semelhante a abstração e a organização da mente do ser humano, consequentemente, as propriedades de O.O. facilitam a construção de AV.

Isto pode ser averiguado com a seguinte correlação, os Objetos Virtuais (OV) são a abstração dos Objetos Reais (OR), os OV e os OR possuem propriedades e comportamentos, e o paradigma O.O. é responsável em encapsular as propriedades e os comportamentos comuns à um conjunto de objetos em uma “classe”.

Além disso, a orientação a objeto tem sido vista como uma poderosa ferramenta de auxílio para promover a reutilização de código e projeto, na quais técnicas de utilização de padrões de projeto (*design patterns*) e *frameworks* viabilizam a implementação de softwares abertos (DEUTSCH, 1989); (GAMMA et al., 1997); (GUIMARÃES, 2001); (JOHNSON&FOOT, 1988); (JOHNSON&RUSSO, 1991) e (JOHNSON, 2001).

### 2.8.1 Análise Orientada a Objeto

Os grandes problemas da análise de sistemas em quase todos os sistemas estão relacionados com: a compreensão do domínio do problema, comunicação dos fatos, evolução contínua e reutilização. O desenvolvedor precisa compreender e modelar o domínio do problema, especialmente no caso de sistemas grandes e complexos. A partir do domínio do problema, concentrar-se-á nos aspectos

específicos do seu problema, ou seja, na descrição das responsabilidades do sistema que está sendo considerado, estabelecendo dessa forma as fronteiras do sistema.

Os requisitos para um sistema são mutáveis. Sabe-se que novos requisitos podem surgir devido às mudanças de software, hardware, legislações, melhorias etc. A identificação de características comuns facilita a absorção de alterações.

Hoje, a reutilização está presente em todo o ciclo de vida do software. O reuso da análise, projeto e código em sistemas semelhantes economiza tempo e recursos. Sabe-se que o domínio de problemas mudou muito pouco nos últimos anos. Os resultados da análise podem ser reutilizados e aperfeiçoados em novos sistemas, com soluções específicas de acordo com as restrições de cronograma e orçamento.

Assim, a AOO aceitou o desafio de, se não resolver, pelo menos amenizar estes problemas de: compreensão do domínio do problema, comunicação dos fatos, evolução contínua e reutilização. Para tal, a AOO baseia-se na aplicação uniforme dos princípios para administração da complexidade de um domínio de problemas e das responsabilidades do sistema dentro dele. Um dos maiores problemas do desenvolvedor é a definição do domínio do problema, que abrange um campo de atividades sob estudo ou consideração. Exemplos de domínios incluem as linguagens, as leis, as finanças e o controle do espaço aéreo.

Definido o domínio do problema, o desenvolvedor se concentrará nos aspectos específicos do seu trabalho, ou seja, na descrição das responsabilidades do sistema que está sendo considerado. Sendo sistema um conjunto ou arranjo de elementos relacionados ou interligados de modo a formar uma unidade ou um todo orgânico (como por exemplo, o sistema rodoviário, de contabilidade ou de irrigação), tem-se que as responsabilidades do sistema podem ser definidas como uma organização de elementos relacionados de modo a formar um todo.

AOO baseia-se em princípios cujo conhecimento é importante para o entendimento dos métodos utilizados no desenvolvimento de sistemas. Estes princípios, usados em diferentes métodos de análise e projeto orientado a objetos, para administrar a complexidade e facilitar a modelagem de sistemas, serão apresentados a seguir.

## 2.8.2 Conceitos de Orientação a Objetos

O conhecimento dos princípios da Orientação a Objetos é de grande importância porque facilita o entendimento das idéias, técnicas e ferramentas deste paradigma. Um método, uma ferramenta, um ambiente ou uma linguagem é tanto mais orientado a objeto quanto mais atende aos princípios da orientação a objeto. Os princípios do paradigma podem variar um pouco de autor para autor, assim vão ser relacionados somente as características de consenso de vários autores e métodos, as vezes a terminologia é diferente apesar de possuir o mesmo conceito.

A terminologia adotada é descrita a seguir, (PRADO,2001):

### 2.8.2.1 Abstração

Abstração é a habilidade de ignorar os aspectos de um assunto não relevantes para o propósito em questão, tornando possível uma concentração maior nos assuntos principais.

A abstração de objetos serve de base para a organização do pensamento e a especificação das responsabilidades do sistema. É a habilidade de descrever novos tipos de dados em termos de seus formatos e serviços que agem sobre eles. Ao aplicar a abstração de objetos, o desenvolvedor define:

- atributos, e
- serviços ou método que manipulam exclusivamente estes atributos.

Atributo é qualquer propriedade, qualidade ou característica que pode ser atribuída a uma pessoa ou objeto. atributo é um dado (informação de estado) para o qual cada objeto em uma classe tem seu próprio valor. Os atributos só podem ser acessados através de um serviço.

Serviço ou método é um comportamento específico que um objeto deve exibir.

Abstração de um objeto é uma classe com um conjunto de atributos e serviços.

### 2.8.2.2 Encapsulamento ou Proteção de Informação

O objetivo do encapsulamento (ocultamento de informações) é restringir o escopo ou visibilidade da informação para obter melhor legibilidade, manutecibilidade e principalmente reutilizabilidade no desenvolvimento de um novo sistema. O encapsulamento permite que o desenvolvedor reuna os aspectos mais instáveis de forma a facilitar sua localização e manutenção em caso de alterações, hoje muito comum nos sistemas.

### 2.8.2.3 Herança

Herança é o mecanismo para expressar a similaridade entre Classes, simplificando a definição de Classes iguais a outras que já foram definidas.

Este princípio permite representar membros comuns, serviços e atributos uma só vez, assim como especializar estes membros em casos específicos.

A herança permite, portanto, a reutilização de especificações comuns, logo no início das atividades de análise. A herança define uma relação entre classes do tipo é-um(a), onde uma classe compartilha a estrutura e o comportamento definidos em uma ou mais classes. O reconhecimento da similaridade entre classes forma uma hierarquia de classes, onde superclasses representam abstrações generalizadas e subclasses representam abstrações, onde atributos e serviços específicos são adicionados, modificados ou removidos. As classes são conectadas por uma estrutura de generalização e especialização, tornando explícitos os atributos e serviços comuns em uma hierarquia de Classes, (PRADO, 2001).

### 2.8.2.4 Polimorfismo

Significa que a mesma operação pode atuar de modos diversos em classes diferentes. Uma operação é uma ação ou transformação que um objeto executa ou a que ele está sujeito. Uma implementação específica de uma operação por uma determinada classe é chamada de método. Como um operador baseado em objetos é polimórfico, pode haver mais de um método para sua implementação.

### 2.8.2.5 Agregação

É o princípio que permite ao desenvolvedor considerar algo muito grande através do enfoque Todo-Parte. Todo-Parte é um dos serviços básicos naturais de

organização dos seres humanos que orienta o desenvolvedor através de um modelo extenso.

Todo-Parte também é conhecido como Agregação, que é um mecanismo que permite a construção de uma classe agregada a partir de outras classes componentes. Usa-se dizer que um objeto da classe agregada (Todo) tem objetos das classes componentes (Parte). Por exemplo, na Figura 9 tem-se que uma casa tem portas, janelas, paredes, escadas e outras partes (PRADO, 2001).

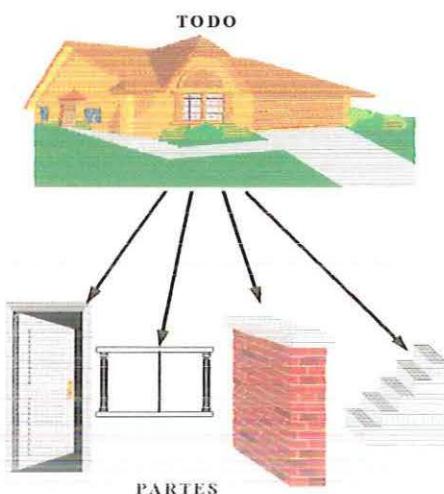


Figura 9. Todo-Parte

### 2.8.3 Linguagens de Programação Orientada a Objetos

Orientação a objetos é uma maneira de pensar nos problemas, usando modelos organizados através de conceitos do mundo real, tal que a construção fundamental são os objetos que englobam estados e comportamento e representam no *software* as entidades do mundo real.

A idéia fundamental de LOO é a utilização e a manipulação de objetos nos programas, ou seja, os objetos são entidades descritas por atributos (dados) e comportamentos (métodos) e as soluções do problema são executadas através do envio de mensagens a objetos.

Existem cinco componentes básicos: classe, instância, objeto, mensagem, e método:

- Classe

- Uma classe de objetos descreve um grupo de objetos com propriedades similares (atributos), comportamento comum (operações), relacionamentos comum com outros objetos e semântica comum.
  - É um tipo de dado (registro) que possui campos que são dados e campos que são funções
  - Uma classe é a descrição de um conjunto de objetos similares, descreve as variáveis de estado e os protocolos de acesso para um objeto daquela classe.
- Objeto
    - Todos os objetos têm identidade e são distintos
    - É uma abstração encapsulada que inclui informações de estado e um conjunto claramente definido de protocolos de acesso.
    - Uma variável de um tipo de classe é um objeto.
  - Instância
    - Os objetos são instâncias de uma classe, as propriedades de qualquer (objeto) são determinadas pela descrição da classe a qual pertence.
  - Método (serviço/função membro) e Mensagem
    - O método define como deve ser implementada uma mensagem para um objeto. Fazem a interface entre o programador e a classe.
    - Mensagem é um símbolo, identificador ou palavra(s) chave(s) especial, com ou sem parâmetros, que representa uma ação a ser praticada pelo objeto.



- Mensagem é qualquer comunicação, escrita ou oral feita entre pessoas[Webster's 77]. Uma conexão de mensagem, em um sistema computacional O. O., modela a dependência de processamento de um objeto, indicando quais métodos (serviços) ele precisa para cumprir suas responsabilidades. As conexões de mensagens existem somente em função dos métodos, e fazem o mapeamento:
  - de um objeto para outro objeto;
  - de um objeto para uma classe (criação de objetos) e
  - de uma classe para outra classe (criação de objetos dentro de outros objetos).
- Numa conexão de mensagem, tem-se uma classe ou objeto emissor que envia a mensagem para uma classe ou objeto receptor para realização de algum processamento. O processamento necessário é nomeado na especificação de serviços do emissor, e é definido na especificação de serviços do receptor.

Resumindo: os objetos são instâncias de classes que responde a mensagens de acordo com os métodos determinados pelo protocolo de descrição de classe, Figura 10.

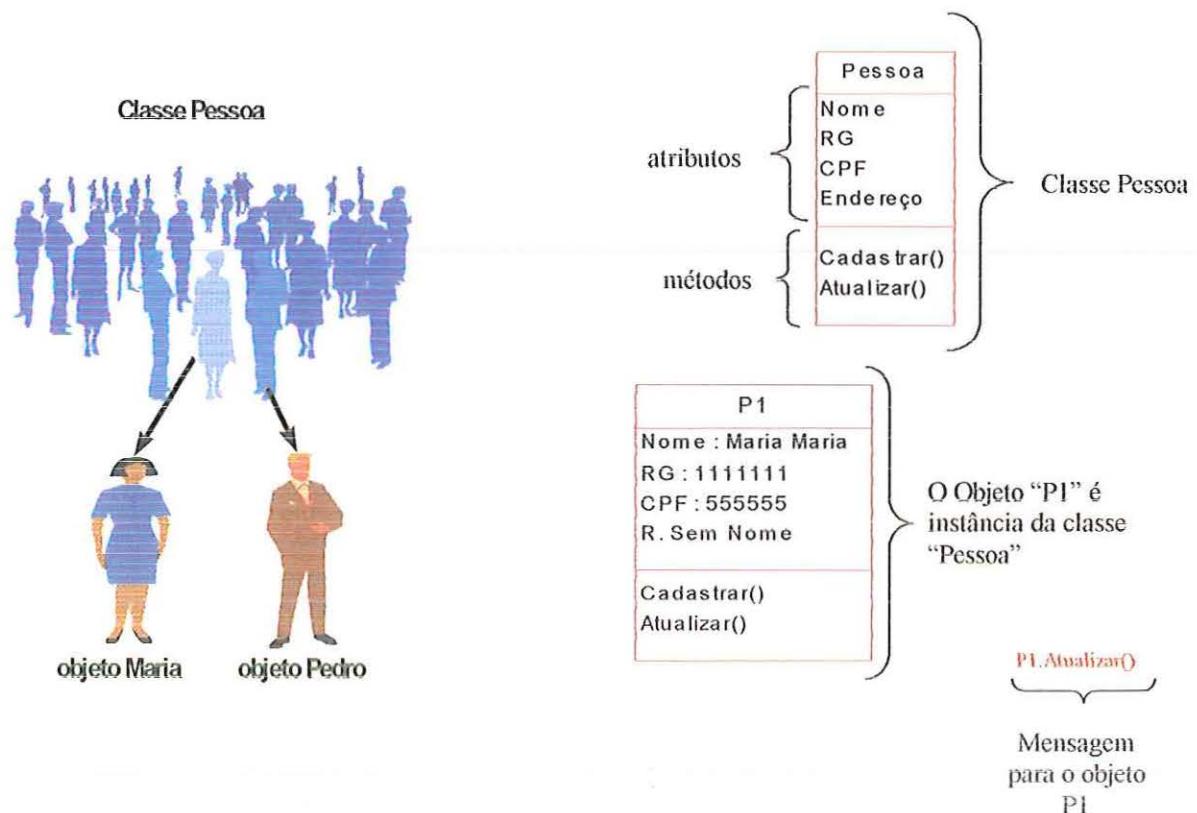


Figura 10. Componentes básicos de LPOO

Para o desenvolvimento de aplicativos orientados a objetos, uma das linguagens disponíveis e mais utilizadas no mercado é o C++ MEYERS (1992) e o STROUSTRUP (1991). Esta pode ser considerada uma linguagem de alto nível, mas também pode trabalhar com códigos de mais baixo nível, na busca de algoritmos mais eficientes.

Existem várias linguagens de O.O., elas devem dar suporte aos conceitos de O.O. visto acima e ainda.

#### 2.8.4 O paradigma de Orientação Objeto em Sistemas de Manufatura

Partindo da perspectiva da Orientação a Objeto, um sistema de manufatura é composto de um número de objetos físicos (i. e. equipamentos de manufatura). Cada objeto físico (exemplo: um torno) tem seus próprios comportamentos caracterizados por métodos (exemplo: tornear) e seus atributos ou estado

torneamento, torneando, fim do torneamento) envolvidos. Um modelo de controle coordena e seqüência a execução de diferentes atividades/métodos de objetos físicos distintos. (WANG, 1996) .

### **3. Proposta de um Metamodelo para Modelagem e Simulação de Sistema a Eventos Discretos, baseado em Redes de Petri e Realidade Virtual.**

#### ***3.1 Introdução***

A simulação requer a criação do modelo de um sistema real, com o propósito de avaliar o seu comportamento sob várias condições. Auxilia o analista a tomar decisões sobre novos sistemas sem precisar construí-los, ou fazer alterações em sistemas já existentes sem perturbá-los. Possibilita aos gerentes visualizar uma operação em várias condições. Permite a análise de interações entre sistemas (integração) e o entendimento de como vários componentes interagem entre si e como afetam o desempenho do sistema como um todo.

A prática da simulação proporciona o discernimento sobre a natureza de um processo para identificar problemas específicos ou áreas problemáticas dentro de um sistema. Além disso, auxilia no desenvolvimento de políticas ou planos específicos para um processo, testando novos conceitos e/ou sistemas antes de sua implementação.

O nível de consistência do modelo está atrelado com a fidelidade com que ele representa o sistema real e todos os resultados de um estudo de simulação são dependentes da qualidade do modelo construído. A simulação do Ambiente Virtual (AV) utilizando Redes de Petri fornecerá a ele maior consistência na descrição do seu comportamento.

### **3.2 Modelagem e Simulação baseada em Redes de Petri**

Este trabalho vai associar os modelos de Redes de Petri aos elementos básicos do paradigma Orientado a Objeto, mais especificamente aos Objetos Virtuais de manufatura, para projetar o controle de Ambientes Virtuais. A RP na construção de AVs vai fornecer as vantagens da modelagem, da análise matemática e da simulação do modelo para fornecer um sistema mais conciso, evitando-se a perda de tempo na partida e na manutenção do mesmo.

O metamodelo é fundamentado na ferramenta de modelagem de Redes de Petri primeiramente devido ao histórico de utilização desta ferramenta nos projetos desenvolvidos no Laboratório de Simulação SEM-EESC-USP, pois ele será agregado a um conjunto de ferramentas de modelagem e simulação baseadas em RP. Além disso, o próprio conceito da RP fornece vários motivos descritos neste tópico.

A Rede de Petri é uma ferramenta de modelagem gráfica e matemática que pode ser aplicada a muitos sistemas. É freqüentemente citada como uma ferramenta promissora para descrever e estudar informações processadas nos sistemas caracterizados como concorrentes, assíncronos, distribuídos, paralelos, não determinísticos e/ou estocásticos (MURATA, 1989). Desta maneira, enquadra-se no contexto deste trabalho, pois os sistemas de ambientes virtuais possuem intrinsecamente tais características.

Conforme RILLO (1987) a Rede de Petri:

- é um método abstrato, o que permite a representação de diferentes tipos de sistemas;
- é um modelo rigorosamente formal (consistência matemática);
- é de fácil aprendizado, funcionando como linguagem de comunicação entre especialistas de diversas áreas;
- possui paralelismo e sincronização;
- representa aspectos estáticos e dinâmicos;
- é um modelo gráfico que facilita o entendimento;

- contém o conceito de estado parcial; e
- possui métodos de análise, inclusive comerciais.

Mas ainda, a RP várias vantagens para o engenheiro de simulação responsável em desenvolver o modelo conceitual do sistema e representá-lo com uma interface em RV, conforme listado a seguir:

- Conforme SMITH (1999) a avaliação formal do sistema RV é desejada, mas normalmente as descrições de AVs utilizarem um nível informal. Assim, a RP tem a habilidade de testar e averiguar um modelo, através das propriedades comportamentais, e métodos de análise, (vide revisão). Estes processos de teste e “validação” auxiliam a verificação da conformidade do modelo em relação ao sistema real.
- A RP tem sido usada com sucesso para modelar, controlar e analisar sistemas a eventos discretos, que são caracterizados pela concorrência dos processos, pelo paralelismo, por serem assíncronos, por possuírem *deadlocks*, conflitos e processos de eventos direcionados, características constantes em Ambientes Virtuais e em sistemas de manufatura;
- A RP é indicada para a modelagem do sistema de controle em sistemas de manufatura, e consequentemente para o controle de um ambiente sintético (um AV).
- A RP tem sido estudada extensivamente nas últimas três décadas, resultando na definição de numerosas extensões e técnicas para análises.
- A RP permite a implementação de análise em tempo real, característica fundamental para realidade virtual.

O modelo de uma RP pode ser extenso, o que dificulta a visualização da lógica e do fluxo do sistema (o percurso das marcas). Entretanto, um modelo extenso de RP para o controle de AV com inúmeros elementos permite estabelecer uma relação direta entre os elementos da RP e os elementos do AV, além de controlar o AV pelo fluxo das *marcas* do sistema de controle em RP.

### **3.3 Interface de RV para a Simulação**

A utilização da RV como interface dá uma conotação um pouco diferente à simulação convencional como mostrado na Tabela 4.

**Tabela 4.** Diferença da simulação convencional com a simulação com interface de RV

Simulação convencional	Os principais interesses se concentram nos resultados e nas análises dos resultados, a visualização (animação) seria um meio de suporte para o entendimento do sistema.
Simulação com Interface de RV	O principal intuito é fornecer o entendimento do sistema, e verificar como o sistema reage às interferências do usuário. Os resultados estão diretamente ligados com as interações efetuadas no ambiente em tempo real.

Assim, há a necessidade de se fazer algumas alterações no processo de desenvolvimento de AV. A tabela 5 compara os passos seguidos pelos desenvolvedores de AV com as propostas do metamodelo.

Tabela 5. Comparação do Desenvolvimento AV com a Proposta

	Desenvolvimento de Ambiente Virtual	Proposta
Modelo do problema	<p>O analista realiza o levantamento de requisitos do problema e o modelo junto ao usuário. Nesta fase é importante identificar as necessidades do problema, atividade que depende muito da qualidade da comunicação entre o usuário (ou engenheiro do conhecimento) e o analista. Na comunicação podem ocorrer ruídos e faltar dados importantes, portanto, quanto maior a dificuldade da comunicação, mais níveis irão existir no modelo espiral do ciclo de vida do software, consumindo tempo e dinheiro.</p> <p>Ponto Importante: O desenvolvimento do sistema está atrelado ao modelo que o usuário descreveu para o analista e da interpretação e implementação do modelo pelo mesmo.</p>	<p>Não existirá o papel do analista de sistemas. O usuário editaria diretamente o modelo em uma ferramenta de modelagem. Existem ferramentas que realizam a averiguação do modelo, testando as suas propriedades e analisando-o.</p> <p>Ponto Importante: A especificação do modelo vai depender do usuário (domínio da ferramenta e seu conhecimento em modelagem) e da ferramenta (capacidade de desenvolvimento de modelos e as ferramentas utilizadas para a “validação”).</p>
Construção do Ambiente Virtual	<p>O ambiente virtual será implementado (codificado) por um analista de sistemas com conhecimento em RV, utilizando bibliotecas de RV para facilitar a programação do ambiente. Uma ferramenta muito útil para a construção e simulação dos mundos virtuais é o editor de realidade virtual, que permite ao projetista verificar imediatamente os resultados da criação ou edição (programação) de objetos simulados.</p>	<p>O ambiente virtual será gerado com objetos virtuais, de manufatura previamente definidos sem a necessidade de codificá-los. Os objetos poderiam pertencer a uma biblioteca de objetos virtuais de manufatura previamente codificados (a codificação dos objetos virtuais é feita conforme as características convencionais acrescentando-se proposições para serem controlados por um modelo). A qualidade gráfica e o realismo do mundo virtual estarão diretamente relacionados com a riqueza da biblioteca.</p> <p>Desde que todos os objetos virtuais necessários para uma simulação estejam disponíveis na biblioteca do metamodelo, não haverá a necessidade de programação.</p>
Integração do Modelo com AV	<p>Cada Ambiente Virtual normalmente possui o seu próprio módulo simulador, que é implementado em paralelo com o ambiente virtual em questão.</p>	<p>A modelagem e a edição de AVs serão separadas e independentes. O sistema deverá ser responsável pela interligação do modelo com o AV.</p> <p>Os eventos do modelo serão relacionados com os métodos dos objetos e os estados do modelo com os valores dos atributos dos objetos.</p>

<b>Execução</b>	<p>Durante a simulação, as entradas do usuário, pelos dispositivos de E/S, são submetidas como eventos ao programa simulador, devendo ser lidas em tempo real para minimizar a latência. Esses dados são usados para atualizar a posição, forma, velocidade, etc, dos objetos virtuais. Tanto a visualização da cena, quanto as outras saídas (som, tato, força, etc.) são fornecidas durante o ciclo de simulação em tempo real. O produto final é um aplicativo executável, ou seja, um ambiente virtual para fim específico. Qualquer mudança no sistema acarreta na necessidade de reprogramar o ambiente.</p>	<p>A simulação com Interface de RV necessitará de um centro nervoso do sistema - o modelo de RP.</p> <p>A dinâmica do ambiente virtual será controlada pelo modelo, pelas trocas de mensagens com os objetos do ambiente virtual.</p> <p>A ferramenta proposta deverá ser flexível e estará relacionada com a seleção do modelo, do ambiente e da interligação. O comportamento do AV será dirigido pelo modelo. As modificações necessárias no AV serão despachadas pelo modelo, ele atualizará constantemente o estado do AV, não exigindo códigos em linguagens computacionais.</p>
<b>Ferramentas</b>	<p>Para facilitar o processo da programação, diversas empresas e algumas universidades produziram sistemas de desenvolvimento de realidade virtual, conhecidos como "VR ToolKits". Esses sistemas são bibliotecas ampliáveis de funções orientadas a objeto voltadas para especificações de realidade virtual, onde um objeto simulado passa a ser uma classe e herda seus atributos inerentes. Isto simplifica enormemente a tarefa de programar mundos virtuais complexos, uma vez que as bibliotecas, sendo ampliáveis, permitem aos projetistas escreverem módulos específicos de aplicações e ainda usar o mesmo núcleo de simulação. Além disso, esses sistemas costumam ser independentes de hardware, suportam alguma forma de conexão em rede, importam mundos virtuais de outros softwares como o AutoCAD, possuem drivers de comunicação com dispositivos convencionais e não convencionais de E/S, suportam alguma forma de iluminação, sombreamento, textura etc. Os sistemas de desenvolvimento de realidade virtual, portanto, ajudam na integração do sistema e no desenvolvimento das aplicações, podendo reduzir substancialmente o tempo de programação. (KIRNER, 1996).</p>	<p>Os projetistas do sistema de simulação de manufatura consumiriam muito tempo e esforço para :</p> <p>Agregar em sua gama de conhecimento as linguagens de programação e a engenharia de ambientes virtuais</p> <p>ou</p> <p>Transferir seu conhecimento técnico de manufatura para os analistas de sistemas desenvolverem os ambiente virtuais a serem simulados;</p> <p>Apesar da evolução do sistemas de desenvolvimento de realidade virtual a programação ainda é a grande barreira para o corpo técnico de manufatura.</p> <p>Então, a proposta é utilizar ferramentas visuais para editar os Modelos, para editar os Ambientes Virtuais com objetos de manufatura predefinidos, e para interligá-los. Com estas ferramentas, o usuário poderá criar e executar simulações com interface de Realidade Virtual sem a necessidade de elaborar códigos de programação.</p>

### 3.4 Metamodelo

A proposta consiste em elaborar um metamodelo para o desenvolvimento de softwares de simulação com interface de Realidade Virtual.

Os softwares produzidos pelo metamodelo forneceram ao usuário final um ambiente que:

- não utilizará linguagem de programação em nenhuma fase;
- editará e verificará o modelo de controle do sistema a ser de simulado;
- definirá o ambiente virtual;
- efetuará a ligação entre o ambiente virtual e o modelo de simulação e
- executará o ambiente virtual iterativo comandado pelo modelo.

Assim, propõe-se um ambiente de modelagem e simulação de sistemas a eventos discretos, um ambiente para edição de ambientes virtuais de manufatura e um ambiente de conexão dos dois primeiros, um ambiente virtual que executa a simulação e um ambiente para controlar e gerenciar a comunicação (Figura 11). A junção destas ferramentas (metamodelo) proporcionará Simulações com Interface de RV conforme a proposta da Figura 12.

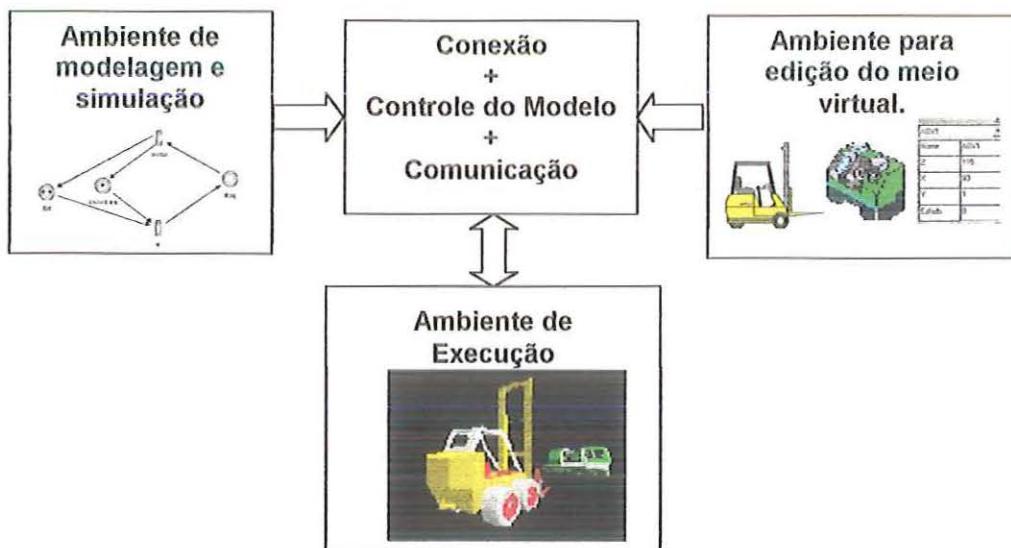
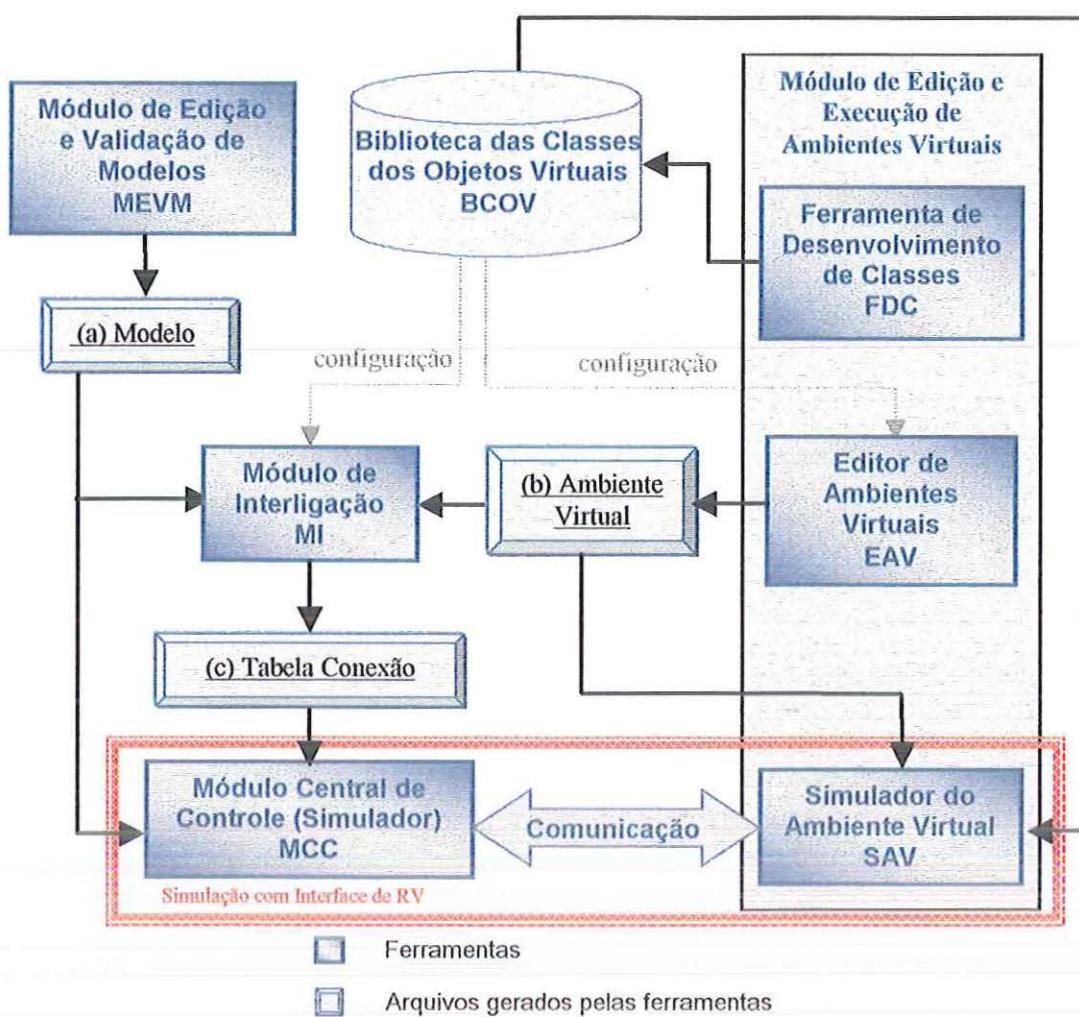


Figura 11. Ambientes do Metamodelo



**Figura 12.** Metamodelo para a Modelagem e Simulação de Eventos Discretos com Interface de Realidade Virtual

Um dos focos do trabalho é a **Simulação com Interface de RV** (retângulo com bordas vermelhas na Figura 12). Ela é composta pelo Módulo Central de Controle (MCC), pelo Simulador de Ambientes Virtuais (SAV) e pelo mecanismo de comunicação entre ele.

As principais atividades são dependentes dos dados do modelo (a), dos dados do Ambiente Virtual (AV) (b) e da Tabela de Conexão (c). Os módulos do Metamodelo deverão estar aptos a criarem os dados (a), (b), (c) da Figura 12 para alcançar a **Simulação com Interface de RV**.

O MCC utiliza os dados do modelo em RP (a) correspondente ao sistema que se deseja simular, os dados vão ser gerados pelo módulo para a edição e validação do modelo (MEVM).

O SAV precisará dos dados do AV (b), os dados serão provenientes do Editor de Ambientes Virtuais (EAV) que permitirá editar informações do sistema desejado.

Os dados do AV (b) junto com Biblioteca de Classes de Objetos Virtuais (BCOV) serão usados pelo SAV para identificar cada objeto e de qual classe ele será instância. O SAV criará em tempo de execução uma coleção de objetos virtuais que irão constituir a cena do AV e irá conduzir a dinâmica do ambiente

Cada classe da BCOV será construída numa Ferramenta de Desenvolvimento de Classes (FDC) com uma linguagem de programação que aceita o paradigma Orientado a Objeto.

Um módulo para realizar a interligação (MI) entre o MCC e o SAV será necessário, pois, a associação irá habilitar a comunicação, o SAV irá entender as mensagens do MCC e vice-versa. A Tabela de Conexão (TC) é o elemento criado pela ferramenta MI, desta forma, os sistemas de simulação são integrados com interfaces em Realidade Virtual.

A arquitetura do metamodelo apresentada pode combinar algumas ferramentas já existentes, e algumas ferramentas que devem ser implementadas.

A fim de entender como cada fonte (classe dos OV, Modelo, AV e Tabela de conexão) será formatada para o metamodelo é necessário observar como funciona a simulação de uma Rede de Petri e como ocorre a simulação de um ambiente virtual baseado no paradigma Orientado a Objetos. A Tabela 6 descreve os pontos relevantes da RP e do AV.

**Tabela 6.** Pontos relevantes do comportamento dos modelos de RP e dos ambientes de RV para o metamodelo

<i>Obs.</i>	<b>Redes de Petri</b>	<b>Ambientes Virtuais</b>
1	Em sistemas de simulação baseados em RP, os elementos são lugares, transições, arcos e marcas.	AVs são compostos por objetos e estes por sua vez são constituídos por atributos e métodos.
2	Os lugares, as transições, os arcos e seus respectivos pesos são parâmetros fixos, definidos antes de iniciar qualquer análise (ou simulação) dos modelos.	Os objetos com os seus respectivos atributos e métodos são definidos antes da simulação do ambiente em Realidade Virtual.
3	O estado da RP em um dado instante é dado por uma marcação M. A marcação M corresponde a um conjunto que relata a quantidade de marcas em cada lugar na rede N em um dado momento ( $M(p_i) \in N$ em $p_i$ )	O conjunto de valores existentes nos atributos dos objetos que compõem o ambiente em um dado momento determina o estado do ambiente.
4	Uma transição t (evento) tem um certo número de lugares de entrada e saída que representam respectivamente, as pré-condições $\bullet t$ e as pós-condições $t \bullet$ de t. Quando as pré-condições da transição são verdadeiras então a transição é dita habilitada e se esta for disparada as pós-condições serão atualizadas.	Os métodos são responsáveis por atualizar os estados dos objetos (mudar os valores dos atributos do objeto) e do ambiente. Quando um método é chamado (o objeto recebe uma mensagem), sua execução causa mudanças no estado do objeto, ou seja, atualiza os valores dos seus atributos.
5	A marca pode se "mover", ou seja, pode ser colocada ou tirada de pontos específicos (lugares); ativa o conceito de condições e eventos.	As mensagens são responsáveis por ativar métodos.
6	Os modelos em Redes de Petri podem possuir ligações externas com elementos reais ou virtuais, ou tão somente simbolizar ações internas ao modelo. Ou seja, uma marca em um lugar do modelo corresponde a um elemento real em um determinado estado (condição), ou ainda, uma transição (evento) do modelo pode corresponder a um acontecimento no ambiente.	O ambiente virtual pode ter pontos de interação com o usuário do sistema, ou seja, pode possuir sensores externos ao ambiente, como por exemplo, uma tecla, um botão, um dispositivo com sensor (HMD, direção, acelerador), uma alavanca, um equipamento etc. A interação efetuada pelo usuário é capturada pelo AV, que atualiza seu estado. O AV pode efetuar atualizações independentes da interação com usuário.

O metamodelo propõe a geração de toda rede em memória e a execução da simulação do modelo com comunicação direta ao ambiente virtual.

Um padrão de comunicação entre o MCC e o SAV será proposto para a comunicação ser estabelecida, integração entre o sistema de simulação e a interface em Realidade Virtual.

Baseando-se nas relações entre a RP e o AV descritas na tabela 6, as próximas seções descreverão as ferramentas para:

- Construção do modelo: Módulo de Edição e Validação de Modelos (MEVM);
- Construção da Biblioteca de Classes de Objetos Virtuais (BCOV): Ferramenta de Desenvolvimento de Classes (FDC);
- Construção do AV: Editor de Ambientes Virtuais (EAV);
- Construção da Tabela de Conexão: Módulo de Interligação (MI);
- Execução da simulação e gerenciamento da comunicação: Simulador de Ambientes Virtuais (SAV) e Módulo Central de Controle (MCC).

### **3.4.1 Módulo de Edição e Validação de Modelos - MEVM**

A dinâmica de um AV pode ser gerenciada por um sistema de controle, responsável por integrar cada objeto virtual ao AV e por controlar todos os eventos do ambiente.

Para o técnico de simulação ou engenheiro do conhecimento os modelos são a base para a síntese de controle tanto para a análise como para a simulação da sua condução. O sistema de controle pode ser definido através de um modelo de RP.

O Modelo em RP para sistemas de controle permite a computação automática de um algoritmo de controle, sendo especificada pela mudança do estado do modelo permitindo ou proibindo os estados seqüenciais, KATO (1999).

Este módulo é essencial para a simulação, pois os modelos podem ser averiguados e devem corresponder ao comportamento do problema real. O modelo produzido neste módulo vai ser utilizado diretamente para controlar o AV, ou seja, os componentes de uma RP vão ser interligados com os elementos de um objeto virtual. A utilização de uma ferramenta automatizada é indicada, pois ela pode fazer a avaliação do modelo a ser simulado para prognosticar a eficiência do sistema e averiguar a sua consistência, visto que, para desenvolver um sistema com RV há um alto custo financeiro e computacional, pois requer tempo considerável de desenvolvimento.

Propõe-se para este módulo a utilização de um editor de RP já existente, que possibilite a edição e "validação" do modelo. Alguns editores foram citados na revisão, e ainda existem muitas outras ferramentas, como pode ser visto em MORTENSEN&CHRISTENSEN (2001), os quais fizeram um levantamento dos vários editores de RP sintetizando as suas principais características se a ferramenta é comercial ou domínio público, se possui representação gráfica (animação) e quais análises executam (vide Apêndice B).

O modelo desenvolvido neste módulo irá ser utilizado pelo Módulo Central de Controle (MCC) que executará o modelo para gerenciar o AV. A gerência do AV ocorre através de trocas de mensagens entre o modelo e o AV.

Propõe-se a utilização da RP ordinária no MCC. Assim sendo, ao desenvolver um modelo de RP para controlar um AV alguns cuidados deve ser tomado para o modelo estar apto a ser interligados ao AV pelo MI. A interligação deve ser efetivada de maneira harmoniosa e deve proporcionar um casamento perfeito entre os componentes de ambos. A próxima seção apresenta algumas recomendações para elaborar modelos que possam ser facilmente interligados.

#### **3.4.1.1 Recomendações para a elaboração de modelos para controle de AVs.**

Existem recomendações para: (a) o desenvolvimento do modelo; (b) análise do modelo e (c) escolha do aplicativo de edição.

- a. Desenvolvimento do Modelo – A modelagem de um sistema de manufatura para controlar um AV difere um pouco da modelagem para controlar a simulação convencional. O AV na metodologia Orientada Objeto é composto de objetos, onde cada objeto possui as suas características e as suas funcionalidades, descritas, respectivamente, pelos seus atributos e métodos. Visto a observação 2 da Tabela 6, verifica-se que é necessário estabelecer uma relação estreita da RP com o AV. Assim, todos os recursos de um Sistema de Manufatura deverão ser modelados de forma unitária, tanto os estados dos recursos quanto as operações que estes recursos efetuam.

Recurso é a denominação dada a todo elemento que compõe o sistema que, de alguma forma, tem a propriedade de modificar o estado de uma determinada variável. Por exemplo:

Recurso é a denominação dada a todo elemento que compõe o sistema que, de alguma forma, tem a propriedade de modificar o estado da uma determinada variável. Por exemplo:

- num sistema bancário, exemplos de recursos são: os funcionários da agência, equipamentos com que os quais eles interagem (computadores, telefones, fax etc.), caixas eletrônicos, máquinas de escrever, etc.
- num sistema de manufatura, exemplos de recursos são: as máquinas ferramenta (tornos, retíficas, fresadoras), dispositivos de manuseio de materiais (robôs, AGVs, empilhadeiras etc.), operadores destes equipamentos, etc.

A figura 13 mostra um modelo em RP que possui 2 Agvs para servir uma máquina. No item (a) o modelo não poderá ser relacionado diretamente com o AV, pois foi modelado mais de um recurso em um único lugar, os dois AGVs são as marcas e assim uma única transição serve os dois AGVs. No item (b) a modelagem para cada recurso AGV, assim este modelo pode ser relacionado diretamente com o AV.

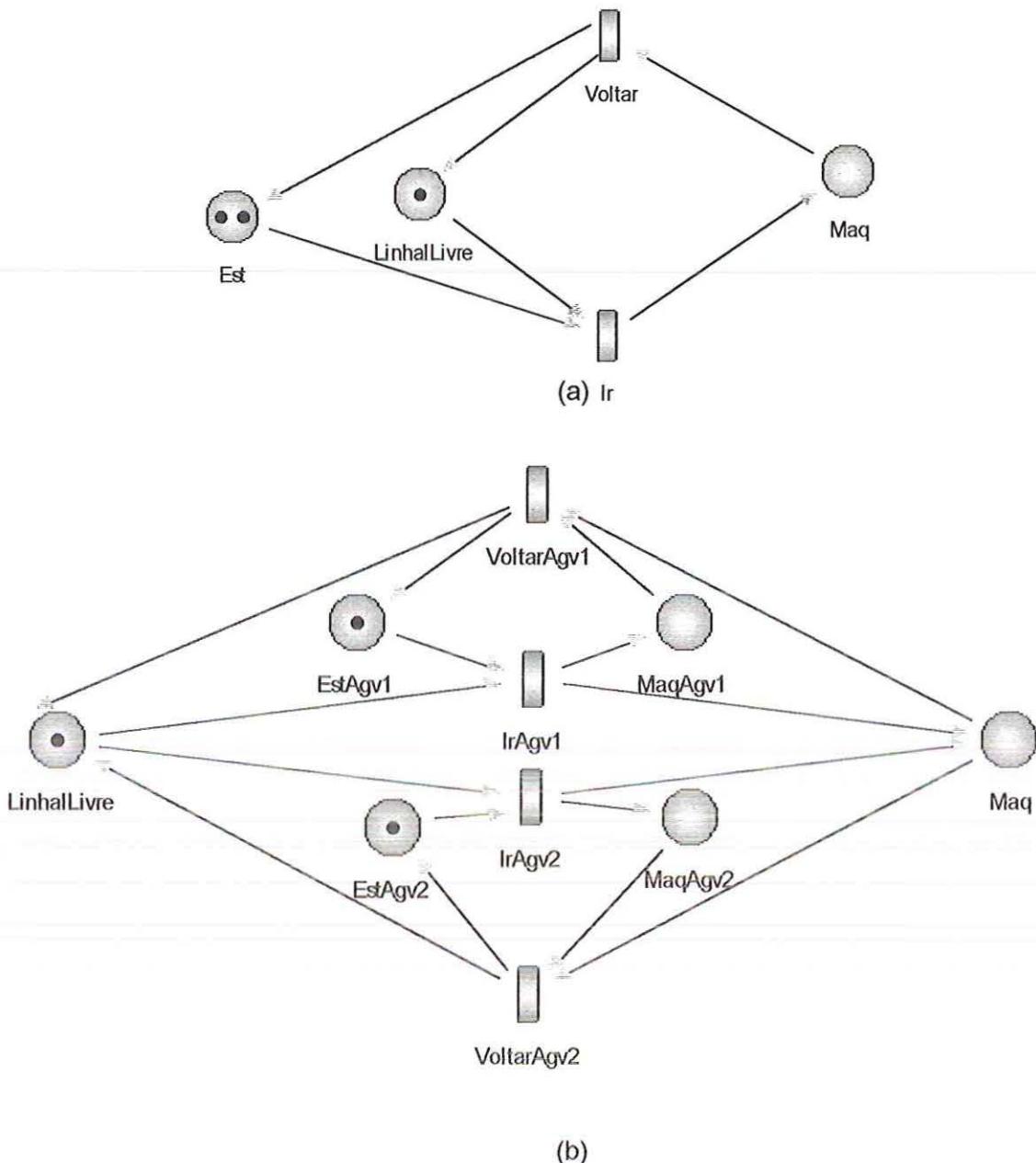


Figura 13. Modelo de um sistema com 2 Agvs para servir uma Máquina

O estado do AV pode ser modificado em tempo real através da interação do usuário. Então, para cada interação do usuário na cena do AV tem que ter pelo menos um lugar correspondente no modelo. O lugar que representa uma interação do usuário normalmente não possui arco de entrada, ou seja, para o lugar  $p_i$  pode-se ter  $\bullet p_i = \emptyset$ . As interações que não mudam o estado do AV não precisam ser representadas no modelo. Por exemplo, as interações de navegação do ambiente não alteram o estado do ambiente, e, portanto, não precisam ser modeladas.

2. Análise do Modelo - As ferramentas de RP normalmente simulam o modelo a partir de uma marcação inicial ( $M_0$ ). Assim, para cada estado inicial em que se deseja fazer uma avaliação do sistema, executa-se novamente o processo de simulação. O estudo de um modelo para controle de AV normalmente irá compreender inúmeros experimentos, pois a colocação das marcas nem sempre se dá pelo disparo de uma transição, mas sim pela interação efetuada pelo usuário no AV. Portanto deve-se utilizar as ferramentas exaustivamente.
- Escolha do Aplicativo de Edição - O aplicativo a ser utilizado para editar o modelo deve possuir algumas características para se enquadrar com a proposta de utilização de RP Ordinária no Módulo Central de Controle (MCC). Uma RP ordinária (Rede de Petri L/T - Lugar/Transição) é um tipo particular de grafo direcionado. Uma Rede de Petri é um grafo direcionado, com pesos nas arestas, bipartido e constituído de dois tipos de nós, chamados de lugares(P) e transições(T), onde arcos (F) saem de um lugar para uma transição ou de uma transição para um lugar. Arcos são rotulados com seus pesos (inteiros positivos) (W).
    - a. O aplicativo deve criar um arquivo para armazenar o modelo. Para o metamodelo manipular e processar as informações de um modelo gerado no Editor de RP é necessário identificar no arquivo a forma descritiva (não gráfica), empregada pelo aplicativo. O aplicativo pode aceitar as extensões de RP e armazená-las, mas deve ser possível extrair do arquivo a estrutura da Rede de Petri ordinária  $N = \{P, T, F, W, M_0\}$ . A RP N extraída deve ser capaz de representar todas possíveis mudanças de estado ( $\sigma$ ) efetuadas pelos disparos das transições utilizando somente elementos descrito por N.
    - b. O Editor pode até ser suprimido, se os dados da rede forem armazenados diretamente em um arquivo. O pré-requisito neste caso é que o modelo deve ser válido.

### **3.4.2 Ferramenta de Desenvolvimento de Classes - FDC**

As classes da Biblioteca de Classes de Objetos Virtuais (BCOV) são desenvolvidas por ferramentas de programação, de maneira que cada instância de uma classe é uma representação em software de um componente presente no ambiente, tais como: o dispositivo mecânico, o torno, a máquina, o sensor, o veículo e assim por diante.

A criação das classes deve seguir o paradigma de Programação Orientada a Objeto (POO), pois:

- Segundo as referências este paradigma representa melhor o mundo real em programas; por esta razão, o desenvolvimento dos elementos de um sistema de manufatura fica intuitivo por sua correspondência ao real;
- os elementos do mundo real podem ser facilmente mapeados em objetos virtuais,
- as características de abstração, herança e polimorfismo resultam em um reaproveitamento de código e deixa o mesmo mais legível;
- a proteção de informação torna mais fácil a manutenção do software, pois as mudanças em uma parte do programa não exigem modificações em outras partes.

As linguagens de POO têm como idéia fundamental criar entidades, encerrando dentro de uma classe tanto os seus dados (atributos) quanto às operações disponíveis sobre estes atributos (métodos), que representam o comportamento da entidade. Os objetos são as instâncias das classes, ou seja, o objeto está para a classe assim como a variável está para o tipo, sendo que a variável não tem a capacidade de representar comportamento e nem características intrínsecas deste paradigma. por exemplo: pessoa é uma classificação e Maria com RG 99999 é uma entidade.

As classes dos objetos virtuais desenvolvidas vão ficar disponíveis na Biblioteca das Classes de Objetos Virtuais. O desenvolvimento das classes de objetos virtuais necessita de um conhecimento técnico na área de computação, pois exigirá conhecimentos de RV e toda extensão do paradigma de Orientação a Objetos (análise, projeto e implementação).

Assim sendo, para desenvolver as classes é necessário domínio dos seguintes itens:

- Análise e projeto Orientado a Objetos para a especificação dos objetos de manufatura, de maneira que os objetos virtuais possuam comportamento correspondente ao real e possam ser visualizados e manipulados pelo usuário;
- Linguagem de programação Orientada a Objetos, como por exemplo, C++ ou Visual Basic;
- Ferramentas de modelagem gráfica para implementação de objetos em 3D, como por exemplo: 3D Studio Max ou AutoCAD ou World Up;
- Desenvolvimento de sistemas de RV; e
- Bibliotecas de RV e/ou bibliotecas gráficas, como por exemplo WorldToolKit, VRML, VTK, etc.

Descreve-se a seguir alguns passos para a construção das classes dos objetos virtuais dos sistemas de manufatura.

#### **3.4.2.1 Caracterização das Classes dos Objetos Virtuais**

O desenvolvimento das classes dos objetos virtuais dos sistemas de manufatura deve seguir as seguintes etapas:

##### *1. Definição dos Objetos Virtuais*

Selecionar objetos que compõem um sistema de manufatura e os objetos existentes no ambiente que influenciam o sistema de manufatura para serem traduzidos em objetos virtuais. Fazer o levantamento das características físicas e lógicas destes objetos a fim de determinar quais as características que serão relevantes para o sistema computacional

##### *2. Objeto Real X Objeto Virtual*

Verificar como cada objeto real se relaciona com o Sistema de Manufatura e com os outros objetos, para verificar a influência deste comportamento em um ambiente virtual.

### *3. Modelagem Geométrica*

A modelagem geométrica abrange a descrição da forma dos objetos virtuais através de polígonos, triângulos ou vértices, e sua aparência, usando textura, reflexão da superfície, cores, etc. Os objetos normalmente são criados usando um modelador 3D, como AutoCad, 3D Studio Max, Modeler do WorldUp, entre outros.

### *4. Modelagem Cinemática*

A modelagem geométrica de um objeto não é suficiente para conseguir uma animação. Os objetos geométricos podem ter comportamentos associados a eles, para isto, deve ser possível agarrar um objeto, alterar sua posição, mudar a escala, detectar colisões e produzir deformações na superfície. A cinemática está relacionada com o comportamento dinâmico do objeto, que é descrito por métodos das classes dos objetos virtuais. As representações gráficas e muitas operações destas classes vão ser relacionadas com transformações geométricas (translação, escala e rotação) e a composição destas, através da utilização de matriz composição.

### *5. Pontos Sensitivos*

Por fim, deve-se identificar os pontos sensitivos dos objetos em relação ao usuário do ambiente.

#### **3.4.2.2 Programação das Classes**

Na programação é indicada a utilização de ferramentas de RV, conhecidas como "VR ToolKits" descrito em Ferramentas de Desenvolvimento de Ambientes Virtuais da Tabela 5.

Os objetos virtuais para o metamodelo deverão possuir algumas peculiaridades, pois serão utilizados para construir um ambiente estático e sem integração entre os objetos. O ambiente "ganhará vida" e os objetos irão trabalhar integrados ao ambiente quando for efetuada a ligação dos elementos do modelo construído no MEVM com o ambiente definido no EAV. Então, as classes têm que estar aptas para esta ligação e para efetivar a dinâmica do ambiente quando a simulação for executada.

Deve-se criar uma classe base que possua as características comuns a todos os objetos virtuais, entre estas, as características que possibilitem ao objeto instanciado fazer a interligação com a RP. Todas as classes devem ser derivadas da classe base.

Para facilitar a criação de classes para objetos virtuais, o programador pode usar os objetos disponíveis ou estender a biblioteca, derivando novos objetos de classes já existentes, de modo a fazer um reaproveitamento de código.

- Métodos dos Objetos Virtuais

Para verificar as necessidades das classes precisa-se conhecer o contexto que elas estão envolvidas. O AV é uma cena composta por um conjunto de objetos virtuais (instâncias das classes de objetos virtuais) e controlada pelo modelo de RP, portanto as classes têm que possuir propriedades para proporcionar a comunicação com a RP.

As classes dos objetos virtuais podem possuir métodos para:

- Métodos para alterar os valores de seus atributos - são invocados quando o estado da RP é alterado, ou seja, quando a rede assume um dado estado  $M_k$ ;
- Métodos para efetuar a dinâmica do objeto - devem efetuar as transformações geométricas, entre outras atividades ligadas ao comportamento dinâmico que seja perceptível no ambiente virtual, ou seja, quando a rede dispara uma transição  $t_k$ ; e
- Métodos para captar as interações do usuário sobre o objeto virtual - quando executados devem enviar mensagens ao controle informando o acontecimento, ou seja, quando o AV recebe um evento do usuário, ele simplesmente remete para o sistema de controle. Este verifica as atividades (transições) que podem ser ativadas no modelo, coloca o *token* no respectivo *place*  $p_k$  e, se alguma atividade for selecionada, requisita a execução do respectivo método ao AV.

Para o metamodelo, o estado do AV só pode ser efetivado a mando do MCC, com exceção dos métodos que captam a interação do usuário. Da mesma

forma, os métodos que estão relacionados com o modelo também só poderão ser executados a mando do MCC. Quando um método é requisitado, o AV responde afetando alguma sensação humana, com eventos visuais, sonoros ou táteis.

Durante a execução da simulação, as mensagens que vão invocar os métodos dos objetos virtuais irão partir do MCC, portanto, quando o AV terminar de executar uma tarefa solicitada, ele deve informar ao MCC. Assim, ao término da execução de um método do objeto virtual que efetua uma dinâmica do ambiente, o AV deve enviar ao modelo um aviso de conclusão da tarefa requisitada pelo modelo.

- *Relação da Classe do Objeto com a Representação da Rede de Petri*

A Biblioteca das Classes dos Objetos Virtuais (BCOV) deverá ser utilizada indiretamente pelo Editor de Ambientes Virtuais (EAV). O EAV é o módulo responsável por configurar o ambiente virtual, informando os objetos a serem instanciados, com as suas respectivas propriedades (valores fornecidos aos atributos da classe). Por este motivo, o EAV terá que conhecer as estruturas dos objetos virtuais (as classes) que vão compor a cena, para poder configurar cada objeto. Da mesma forma, Módulo de Interligação (MI) precisará conhecer as estruturas dos objetos virtuais para relacionar o AV com a RP.

Assim, é proposto que o projetista das classes tenha em sua documentação, e também na forma de arquivos, a descrição de cada classe da BCOV. O arquivo deve possuir um formato que facilite as tarefas de edição do ambiente e também de ligação entre o AV e a RP. Estes arquivos serviriam como configuradores da entrada de dados que o usuário precisa fornecer ao sistema.

Relembrando, através de um módulo de interligação (MI), os atributos e os métodos dos objetos virtuais vão ser relacionados com os lugares e transições da Rede de Petri. Para padronizar-se o formato da estrutura dos objetos virtuais, deverá ser adotada a seguinte regra dentro da própria classe: um atributo, que pode assumir somente um valor dentro de um conjunto de valores finito em um dado momento, deve ser um tipo enumerado (Tabela 7).

**Tabela 7.** Exemplos de tipos enumerados

Tipos de Atributos	Valores que podem ser atribuídos ao atributo
DiaSemana	Segunda, Terça, Quarta, Quinta, Sexta, Sábado e Domingo.
EstadoBotão	Pressionado, NãoPressionado.
Vazio	Verdadeiro, Falso.

Esta regra é necessária para dar a capacidade de se efetuar a ligação de vários *lugares* da RP a um mesmo atributo de um OV. A relação do lugar poderá ser com o valor do atributo, assim um *lugar* com marca significa que o atributo no AV possui um determinado valor, definido diretamente na ligação do AV com a RP.

### **3.4.3 Editor de Ambientes Virtuais - EAV**

A tarefa de criar mundos virtuais complexos por meio de um editor e de uma biblioteca de objetos virtuais ampliável simplifica enormemente o trabalho dos projetistas de simulação.

Este módulo é a interface para a criação do ambiente, não representa a lógica e nem a dinâmica de execução, mas sim a descrição dos objetos pertencentes ao ambiente e seu estado inicial.

Assim, deve-se implementar um editor gráfico para construir os ambientes virtuais com objetos derivados da BCOV. O AV que a ser criado neste aplicativo deverá ser associado com um modelo de RP através do MI, para possibilitar a execução do ambiente virtual. Para que seja possível elaborar a associação com o modelo, o Editor de Ambientes Virtuais deve gerar um arquivo de dados constando todos os objetos que devem ser instanciados (criados) pelo Simulador de Ambiente Virtual (SAV), com os valores iniciais associados aos respectivos atributos dos objetos.

A tarefa de edição do AV irá ser semelhante a criação de editores gráficos orientados objetos ou de editores de ambientes gráficos de simulação sem a ferramenta de execução da simulação (ex. AutoMod®, ProModel® e ARENA®), ou

até mesmo editores de Ambiente Virtuais como WordUp sem a parte de execução do ambiente.

O EAV possuirá uma interface gráfica para a manipulação de objetos, como a criação, remoção e posicionamento e ainda irá utilizar caixas de diálogo para caracterizar os objetos do ambiente. Os objetos são representações gráficas das classes existentes na BCOV. Uma biblioteca completa, com todos os elementos de um sistema de manufatura, permite construir ambientes virtuais utilizando-se somente o editor, sem programação.

O EAV gera um arquivo de dados sobre os objetos que constituem a cena (os objetos com os seus atributos - características e/ou estado dos objetos). Assim, pode-se alternativamente construir um ambiente virtual criando-se um arquivo com a descrição dos objetos que compõem a cena, com os valores iniciais do ambiente virtual, dispensando o uso do editor.

O editor será uma ferramenta sujeita a atualizações constantes, pois a biblioteca de objetos virtuais pode crescer ou sofrer modificações com acréscimo de novos elementos de manufatura e/ou modificando os já existentes. Por isso neste sub-módulo é recomendada a utilização de técnicas de orientação a objetos que conduzam ao desenvolvimento de componentes reutilizáveis de software.

Os componentes do programa precisam ser *projetados para reutilização*. Para alcançar esta meta, o foco é utilizar as técnicas básicas de orientação a objetos dentro de dois conceitos mais avançados na área: o conceito de *frameworks* e o uso de padrões de projeto (*design patterns*).

*Framework* é uma técnica de reutilização de código. A técnica consiste em criar uma aplicação semicompleta, reutilizável, que pode ser especializada para construir aplicações que satisfaçam requisitos específicos. Esta especialização é chamada de instância, conforme a Figura 14.

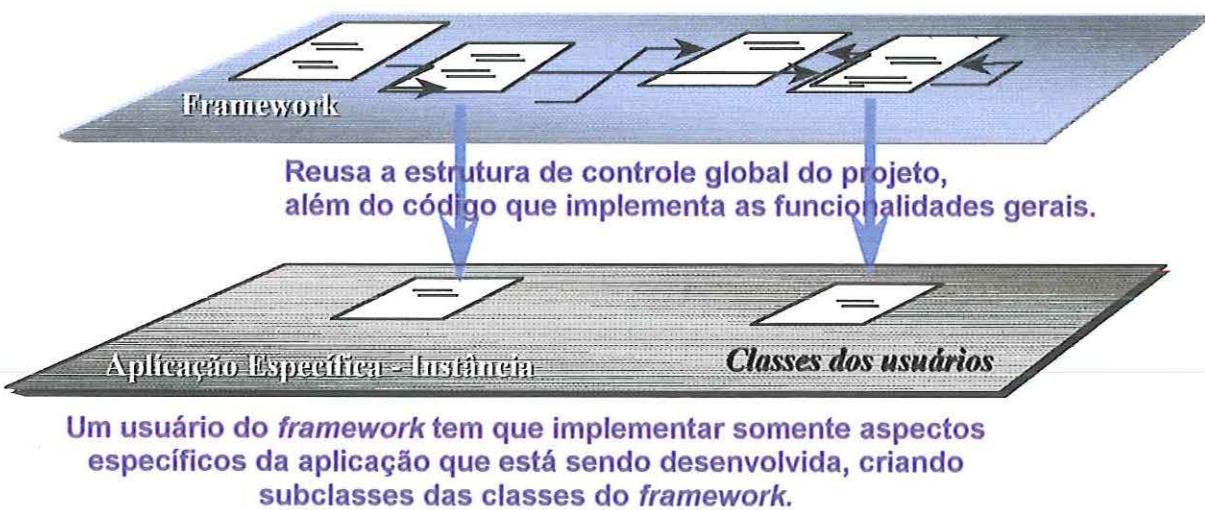


Figura 14. Estrutura de um *framework*

Formalmente, um *framework* é um conjunto de classes que encerra um projeto abstrato para soluções de uma família relacionada de problemas e suporta reutilização em uma granularidade maior do que as classes. Durante as fases iniciais da história do sistema, um *framework* faz um uso mais efetivo de herança e o engenheiro de software precisa conhecer como o componente é implementado para poder reutilizá-lo. Quando o *framework* torna-se mais refinado, ele fornece componentes como “caixas-pretas” que podem ser reutilizados sem conhecer suas implementações, abstraindo a complexidade do seu funcionamento. (JOHNSON, 2001).

Padrões de projeto podem ser entendidos como boas soluções para problemas encontrados no desenvolvimento de sistemas que são documentadas para indicar como abordar estes problemas no projeto e implementação de novas aplicações. Cada padrão de projeto descreve como solucionar uma pequena parte de um problema de projeto maior. O uso de padrões de projeto no desenvolvimento de sistemas tende a uma implementação com código mais enxuto e de maior qualidade, além de evitar o desperdício de tempo reinventando soluções conhecidas. (GAMMA, 1997).

Padrões de projeto e *frameworks* são utilizados de forma combinada, onde os padrões de projeto documentam o propósito e o *framework* o uso. Com o uso adequado das técnicas descritas durante as fases da construção do sistema, os componentes do sistema serão projetados e implementados antecipando sua reutilização e facilitando um ciclo de vida evolutivo para o software, com bases

sólidas para evitar ou reduzir retrocessos no decorrer do desenvolvimento e manutenção do sistema.

### **3.4.4 Módulo de Interligação - MI**

A RP é amplamente defendida como uma ferramenta que dá suporte, ao projeto e validação de modelo de ambientes de realidade virtual, deixando-os confiáveis. Vários trabalhos utilizaram a RP para verificar o modelo do ambiente virtual, mas codificando a RP em uma linguagem de programação que suporta RV, tarefa que impossibilita a generalização do uso.

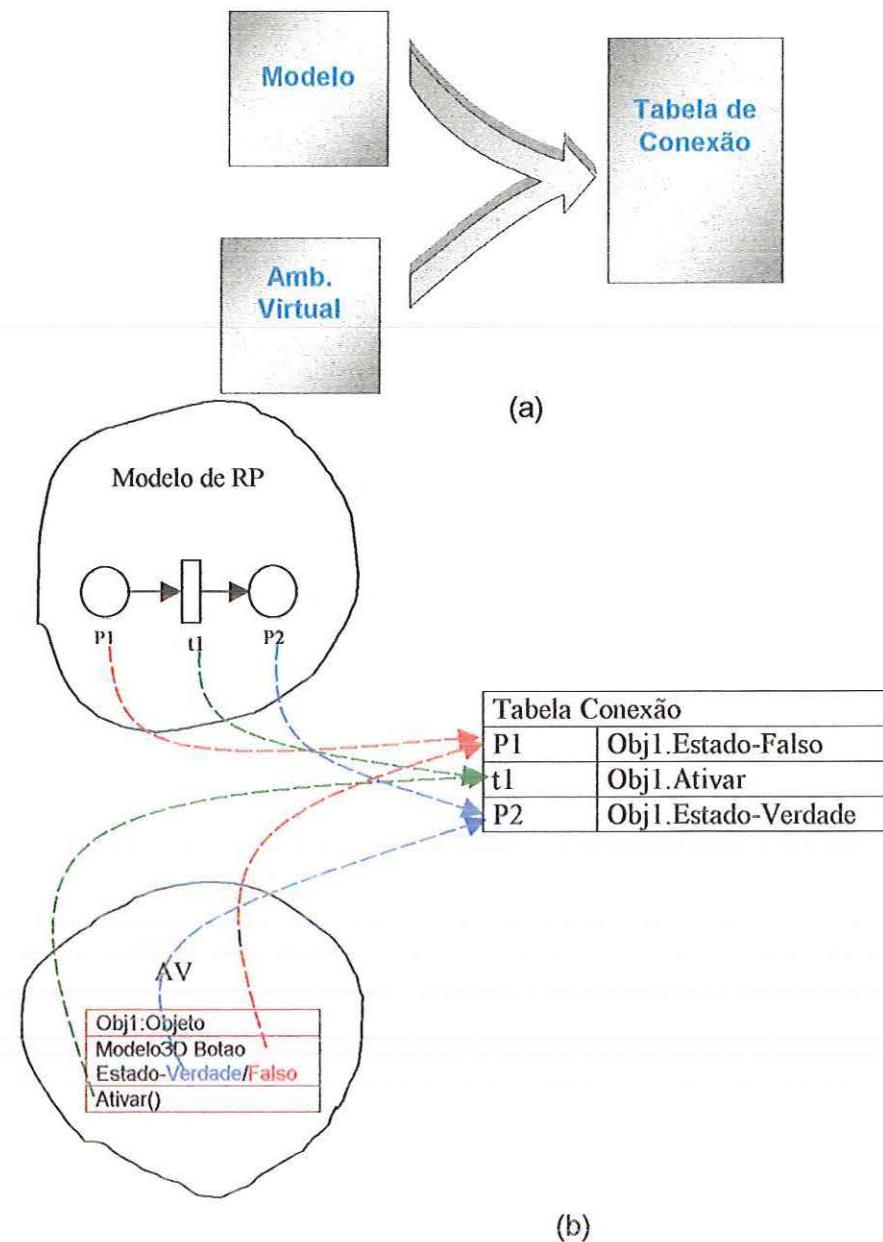
VALÉRIO,<sup>98</sup>, transformou todo o modelo de Redes de Petri do intertravamento de um torno em fluxogramas e codificou-os em linguagem de programação para controlar o protótipo virtual do torno. MASCARENHAS,<sup>98</sup> fez o inverso. Pegou o código do sistema de um CAVE, elaborou o modelo em RP, editou-o no Editor de RP "Cabernet toolset" e fez inúmeros experimentos para analisar o modelo, chegando a encontrar um erro neste consagrado sistema, afinal, a Universidade Illinois foi a percussora nesta forma de imersão no ambiente.

Para os problemas apresentados, a RP produziu bons resultados para a modelagem do controle do AV, mas isto não eliminou a onerosa tarefa de programar Ambientes Virtuais.

O metamodelo apresentado nesta tese visa cobrir os esforços de programação. Para isto, foi estabelecido um formato para as classes dos objetos virtuais (item 3.4.2), e como deve ser o modelo (item 3.4.1). O Módulo de Interligação será responsável em "celebrar o casamento" do AV com o modelo - Figura 15 (a).

A interligação somente é efetuada quando há dados suficientes do modelo em RP e do AV. Os dados relevantes para a ligação do modelo são os lugares e as transições. Os ambientes virtuais são os objetos que compõem o ambiente, com os respectivos métodos e pontos que podem ser ativados (atributos com um determinado valor).

O módulo de interligação ligará os lugares da RP com os valores dos atributos caracterizados como pontos ativáveis do AV e liga as transições da RP com os métodos do AV, criando uma tabela de conexão, Figura 15(b)



**Figura 15.** Tabela de Conexão realiza a conexão do modelo com AV

A tabela de conexão servirá para identificar cada elemento dos objetos que podem ser ativados ou atualizados em tempo de execução. As classes dos objetos virtuais não serão necessárias, mas as estruturas das classes têm que estar presente, pois quando o MI identifica a classe de um objeto declarado no EAV, ele precisará saber quais as características e comportamentos deste objeto para fazer o relacionamento.

### **3.4.5 Ferramentas para Executar a Simulação e Gerenciar a Comunicação**

Para estabelecer a comunicação entre Ambiente Virtual e a Simulação será necessário criar um padrão de comunicação para enviar e receber mensagens entre o sistema de simulação (MCC) e o simulador de ambiente Virtual (SAV).

O MCC é um aplicativo que contém um algoritmo que trata a dinâmica das RPs e SAV é um aplicativo que representa um universo virtual vazio, com alguns objetos padrão, como luz e ponto de vista (view point), e possui a capacidade de criar objetos virtuais dinamicamente. Os objetos que o SAV pode criar devem obrigatoriamente ser instâncias de alguma das classes da BCOV.

Analizando as associações entre a RP e a Metodologia O.O. adotada para o desenvolvimento de AV, apresentadas na tabela 6, foi possível identificar as seguintes relações:

- Um Lugar do sistema de simulação deveria ter ligação direta com um determinado valor de um atributo do objeto do ambiente de Realidade Virtual;
- Nem todos os lugares precisam ter correspondentes no ambiente virtual, mas todas as iterações do usuário no ambiente devem ser representadas no modelo;
- Os objetos virtuais podem mudar de estado com a iteração do usuário. Exemplo disso são os botões, alavancas, ou operações mais complexas que podem acionar algum método no AV como mover, ligar ou apagar, dentre outros. Os eventos do AV decorridos da iteração do usuário deverão possuir relações diretas com a colocação da(s) marca(s) no(s) lugar(s) do modelo em Rede de Petri;
- Alguns lugares e transições não possuem nenhuma ligação direta com pontos virtuais, mas são necessários para que se faça uma modelagem coerente do sistema que se pretende simular; e o inverso também pode ser válido, objetos virtuais que não possuem ligação com o modelo, mas podem estar no ambiente para melhorar a visualização do mesmo.

- nem todas as transições precisam ter correspondentes no ambiente virtual, mas todas as ações dependentes de análise devem estar representadas por transições;

Com os pontos listados na tabela 6 e a análise feita acima foi possível estabelecer a comunicação da **Simulação com interface de RV**, assim:

- efetivar um evento de iteração no ambiente virtual simboliza a colocação direta de uma marca em um lugar e desta forma todas as ativações devem gerar mensagens para o sistema de controle, (as marcas serão colocadas em lugares que normalmente não possuem transições que os precedem);
- O AV ao concluir uma ação avisa ao seu correspondente na RP;
- quando uma transição está habilitada e vai ser disparada, a RP tem que avisar o seu correspondente no AV,
- quando a quantidade de marcas em um lugar for alterada, a RP tem que avisar o seu correspondente no AV,
- transições internas que são disparadas ou lugares que recebem marcas, mas não possuem correspondentes externos, não serão enviados ao AV, mas sim tratados internamente;

A Figura 16 sintetiza as principais atividades do MCC e do SAV em relação aos pontos listados. O relógio (⌚) da figura significa que atividade vai ser realizada de tempo em tempo com intervalo pré-determinado.

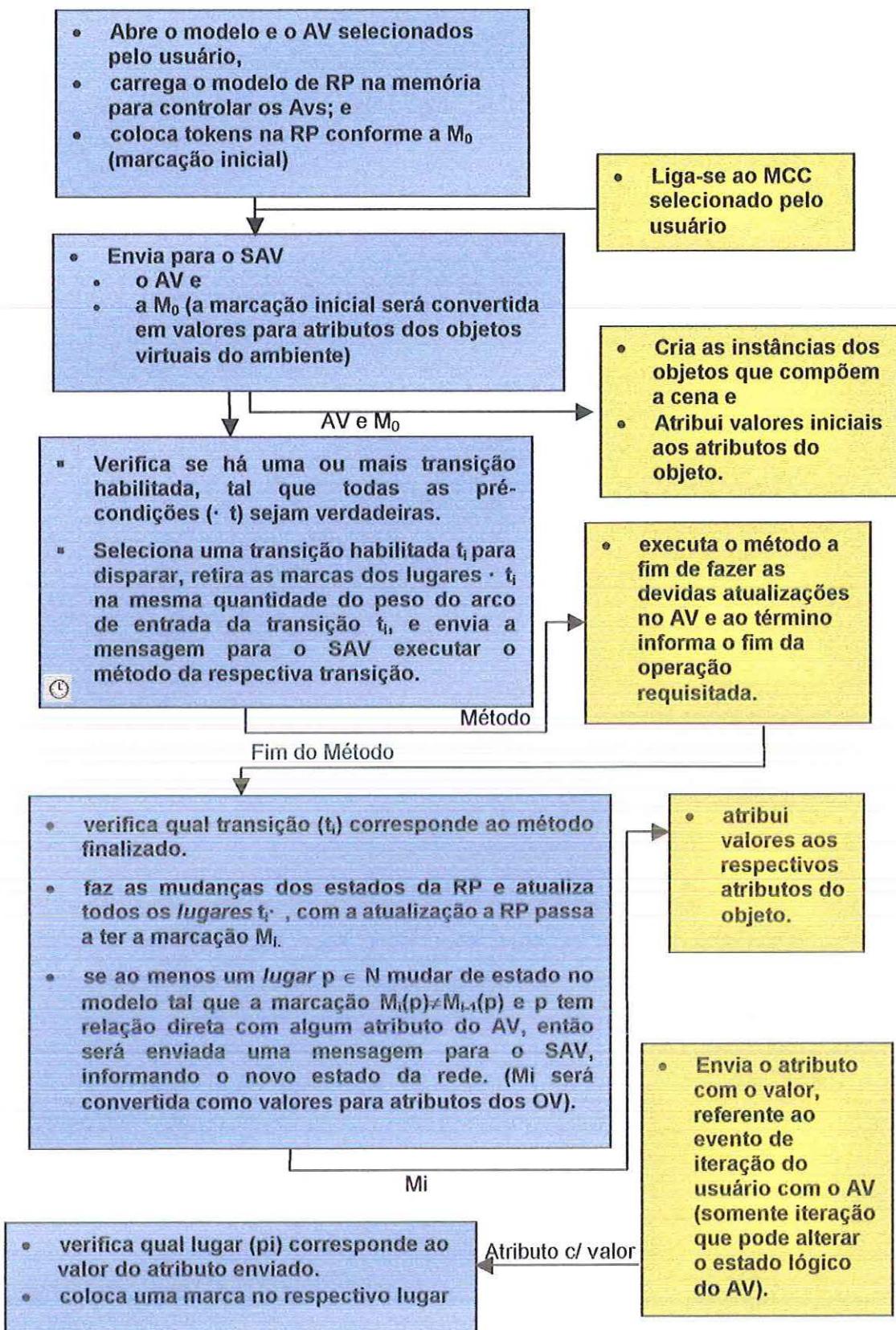


Figura 16. Atividades do Módulo de Central de Controle – MCC e do Simulador de Ambiente Virtual - SAV

Para efetuar as equivalências do mundo virtual com o modelo, o MCC utilizará a tabela gerada no MI, que relaciona o elemento do mundo virtual com o seu correspondente no modelo.

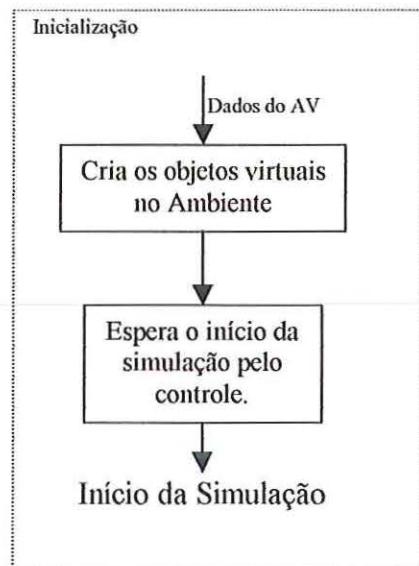
### 3.4.5.1 Simulador do Ambiente Virtual -SAV

Para construir o SAV, também deve-se utilizar o paradigma Orientado a Objetos para a construção de um *framework*, pois ele deverá ser constantemente atualizado para agregar novos objetos virtuais, ou modificar os já existentes, acrescentando novas características e/ou novos dispositivos de interação.

O SAV é um aplicativo que vai ser construído para compor e controlar AVs. Em outras palavras: o SAV é um universo que possui a capacidade de criar dinamicamente qualquer objeto desde que seja instância das classes contidas na BCOV. Possui os elementos vitais de um ambiente como luz ambiente, ponto de vista (equivale a uma câmera no mundo real, possibilita especificar que o ângulo o AV será visto), objetos sensores (capta a entrada de dados provindos de dispositivos externos). O MCC é responsável em gerenciar o universo por meio do modelo de RP, portanto o SAV será incumbido de:

- receber e distribuir as mensagens aos objetos virtuais e
- enviar mensagens ao MCC.

Para fazer a simulação de um ambiente primeiramente deve-se receber a configuração do AV que vai ser simulado, conforme a Figura 17.



**Figura 17.** Configuração do Simulador de Ambientes Virtuais

O SAV deverá ficar monitorando constantemente a simulação, logo após a configuração do ambiente. A monitoração identifica as tarefas do SAV durante a simulação (Figura 18). As tarefas conforme mostrado na Figura 18 estarão dentro de um *loop*, até que o servidor finalize o controle.

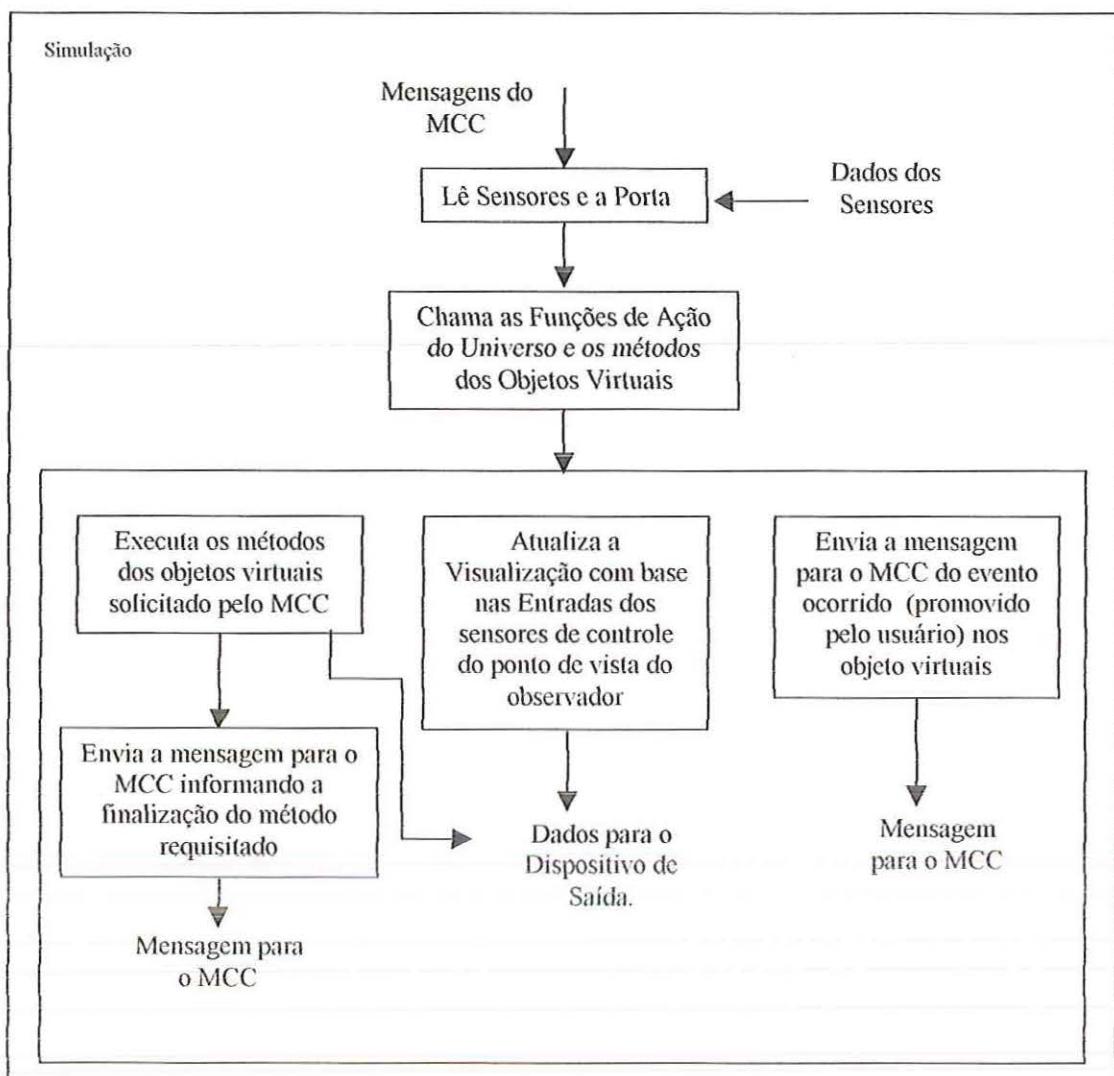


Figura 18. Monitoramento do Ambiente Virtual

O SAV é responsável pela visualização 3D e pela dinâmica do ambiente virtual.

Durante a execução da simulação, o MCC envia mensagens para os métodos dos objetos virtuais que estão no SAV, os métodos podem modificar o modelo 3D associado, mudando sua forma, posição ou orientação.

O ambiente virtual pode ter pontos de iteração com o usuário do sistema, ou seja, possui sensores externos ao ambiente. A entrada é usada para manipular o comportamento do objeto virtual e o objeto deve possuir um método que identifique o evento produzido pelo dispositivo, mas reage enviando uma mensagem para o controle (MCC). Em outras palavras, a iteração entre o usuário e o sistema simulado ocorrerá indiretamente, isto é, o usuário interage com os objetos que vêm no

AV monitorado pelo SAV e este envia mensagens para o MCC. A mensagem é tratada somente dentro do MCC.

Porém, os eventos enviados pelos dispositivos para controlar o ponto de vista do observador para efetuar a navegação no ambiente não é controlada pelo MCC, pois a navegação não altera o estado lógico do sistema de simulação. O controle da navegação é responsabilidade do SAV.

### 3.4.6 Módulo Central de Controle - MCC

O Módulo Central de Controle tem como finalidade centralizar todo o sistema de controle do(s) Ambiente(s) Virtual(is), controlar e gerenciar a comunicação. (Figura 19)

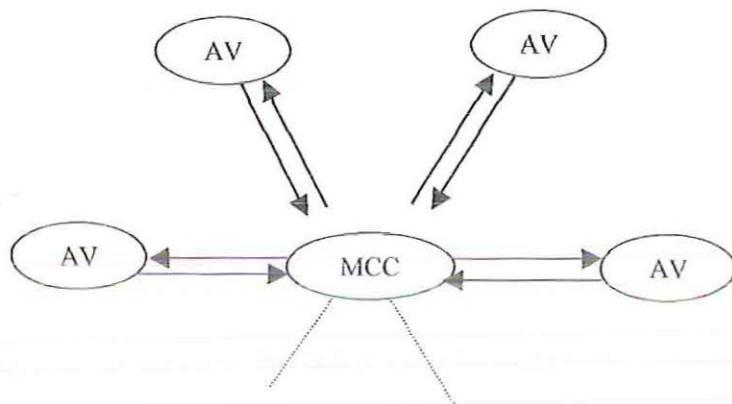
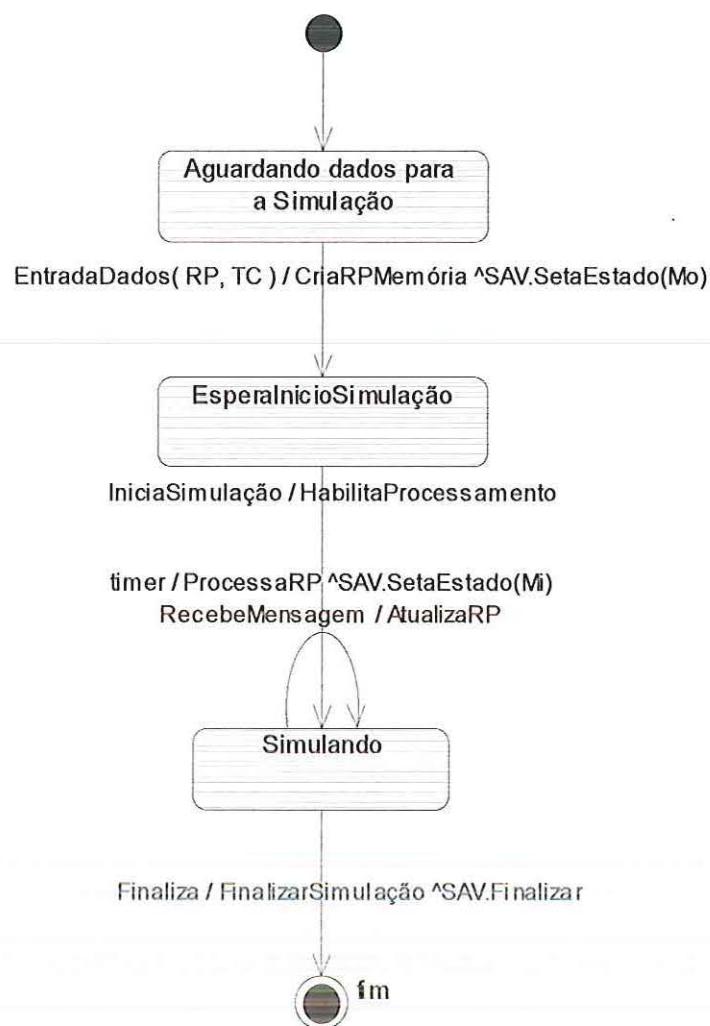


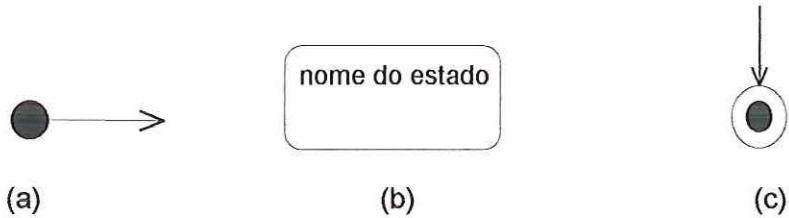
Figura 19. Módulo Central de Controle

A comunicação entre o AV e o MCC será monitorada freqüentemente. Com o Diagrama de Estado da Figura 20, é possível verificar a funcionalidade geral do MCC, e como ele efetua a comunicação e o gerenciamento.



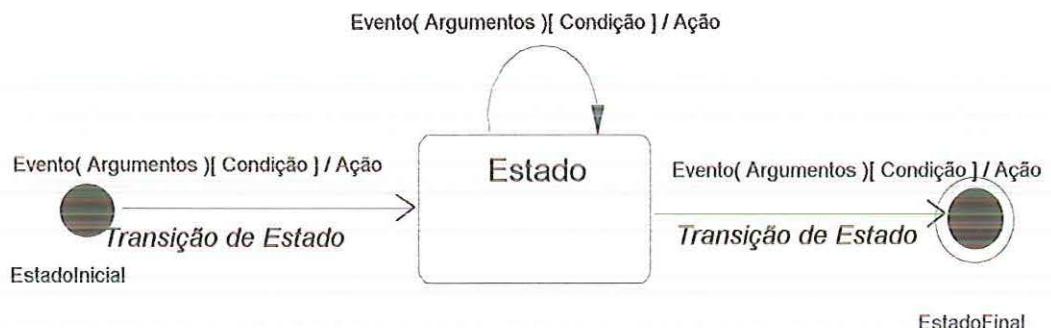
**Figura 20.** Diagrama de Estado do MCC

O Diagrama de estados da figura 20 é um grafo direcionado de estados conectados por transições que mostra um estado inicial, um ou mais estados finais e as transições de estados entre eles. Um estado inicial mostra o início do diagrama. Em cada diagrama de estado há somente um estado inicial. A notação gráfica de um estado inicial é mostrada na Figura 21(a). Um estado é uma condição durante a vida de um objeto. A notação gráfica do estado é mostrada na Figura 21(b). O Estado final representa o término de um sistema. A Figura 21 (c) mostra a notação gráfica de um estado final.



**Figura 21.** Estado Inicial (a), Estado (b), Estado Final (c).

A transição de estado pode ser rotulada com o evento e opcionalmente com uma ação separado do evento por uma barra como se segue: evento/ação. Se ocorrer a transição e quando ocorrer, a ação especificada é executada instantaneamente. Algumas ações simplesmente geram um evento, elas podem também causar outros efeitos como: modificar valores de condições e itens de dados, iniciar ou parar atividades, entre outros. A sintaxe adotada é: NomeDoEvento(ListaDeArgumentos) [Condição] / Ação conforme a Figura 22.



**Figura 22.** Transição de Estado

Através destas características, é possível estabelecer um formato de comunicação do MCC.

Quando o AV possuir o seu correspondente na RP e vice-versa:

- O modelo de RP deve informar ao AV as transições que foram disparadas;
- A RP receberá uma mensagem após a finalização de uma operação no ambiente virtual. No modelo isto indicará que foi concluída a transição  $t$  e, como consequência, tem-se que atualizar o estado do modelo (colocar marcas nos lugares  $t\bullet$ ),
- Na RP as marcas dos lugares informam os estados dos objetos, se uma máquina está em funcionamento ou parada, se uma luz está apagada ou acesa, se AGV está no ponto x ou em movimento, quantos AGVs chegaram no

estacionamento, portanto o modelo de RP deve informar ao AV quando um lugar é atualizado (mudança na quantidade de marcas), ou seja, o ambiente virtual deverá receber uma mensagem com valores para os atributos de seus objetos.

- A RP recebe a informação que mudou algum estado do AV, isto significa que ocorreu um evento de iteração do usuário com o AV, assim no modelo de RP deverá colocar marcas nos lugares que representam a iteração na RP.
  - " por exemplo, ao apertar um botão do teclado, a RP será comunicada e colocará uma marca no lugar que correspondente ao botão, o controle em RP verifica se tem condições de atender o pedido do usuário, ou seja, verifica se a colocação de marca(s) no(s) respectivo(s) lugar(s) ativou alguma transição, caso positivo o MCC envia uma mensagem requisitando a execução de um método.

O MCC monitora constantemente o modelo para verificar se alguma transição está apta a ser disparada e se alguma marca chegou do AV para ser colocado no modelo.

### ***3.5 Considerações Finais do Metamodelo***

O metamodelo direciona o desenvolvimento de sistemas computacionais para a simulação com interface de RV. Um aplicativo computacional de simulação, criado seguindo o metamodelo, proporciona aos seus usuários um meio para desenvolver projetos de simulação de mundos virtuais iterativos sem muito esforço em programação de RV, e ao mesmo tempo um AV mais consistente por utilizar uma técnica fomial.

Os resultados da implementação seguindo o metamodelo dependerão de quanto e de como cada tecnologia (simulação e modelagem, RV, Sistemas Distribuídos de RV - comunicação, Orientação a Objeto, frameworks) será empregada, quanto melhor o aproveitamento do potencial de cada tecnologia melhor será a capacidade do aplicativo de simulação de RV.

## **4. Aplicação do Metamodelo para a Implementação de um Sistema Computacional de Simulação com Interface de Realidade Virtual para Sistemas de Manufatura**

### **4.1 Introdução**

Este capítulo descreve o desenvolvimento de um protótipo - software para a simulação de sistemas de manufatura com interface de Realidade Virtual (RV) - seguindo o metamodelo proposto no capítulo 3.

*São abordados neste capítulo:*

- Os recursos necessários para a implementação,
- Sugestão de aplicativos existentes,
- Os passos da implementação de cada aplicativo,
- O desenvolvimento de sistemas de simulação com os aplicativos propostos no metamodelo e
- Um estudo de caso da simulação de um sistema de rotas de AGVs por meio dos aplicativos desenvolvidos.

## **4.2 Recursos Físicos**

Os recursos básicos necessários para o desenvolvimento de um sistema de simulação utilizando o metamodelo, são:

- Sistema Computacional (*hardware + software + pessoas*),
- Um aplicativo para edição e validação de Redes de Petri,
- Biblioteca de Realidade Virtual e/ou Biblioteca Gráfica,
- Modelador Geométrico tri-dimensional (3D) e
- Uma ou mais linguagem de programação com o respectivo compilador, preferencialmente linguagens voltadas a POO.

Alguns recursos podem facilitar o desenvolvimento de sistemas e agilizar as tarefas a serem desempenhadas, por exemplo: um processador mais potente, mais memória para a placa mãe, placa aceleradora de vídeo, até softwares com elementos básicos previamente construídos e CASEs.

O usuário do ambiente poderá ver os modelos 3D dos objetos através de dispositivos especiais de Realidade Virtual (RV) ou através de dispositivos convencionais de uma mesa de trabalho e será capaz de iterar com os objetos, usando dispositivos como luvas, *trackers*, teclado ou mouse.

## **4.3 Implementação**

### **4.3.1 Edição do Modelo e a Validação**

O modelo deve ser construído numa ferramenta com funcionalidades para criação, manipulação e simulação. Neste trabalho foi adotado o Editor de Redes de Petri - Petri Net Tools (Figura 23), desenvolvido no Laboratório de Simulação da SEM-EESC-USP.

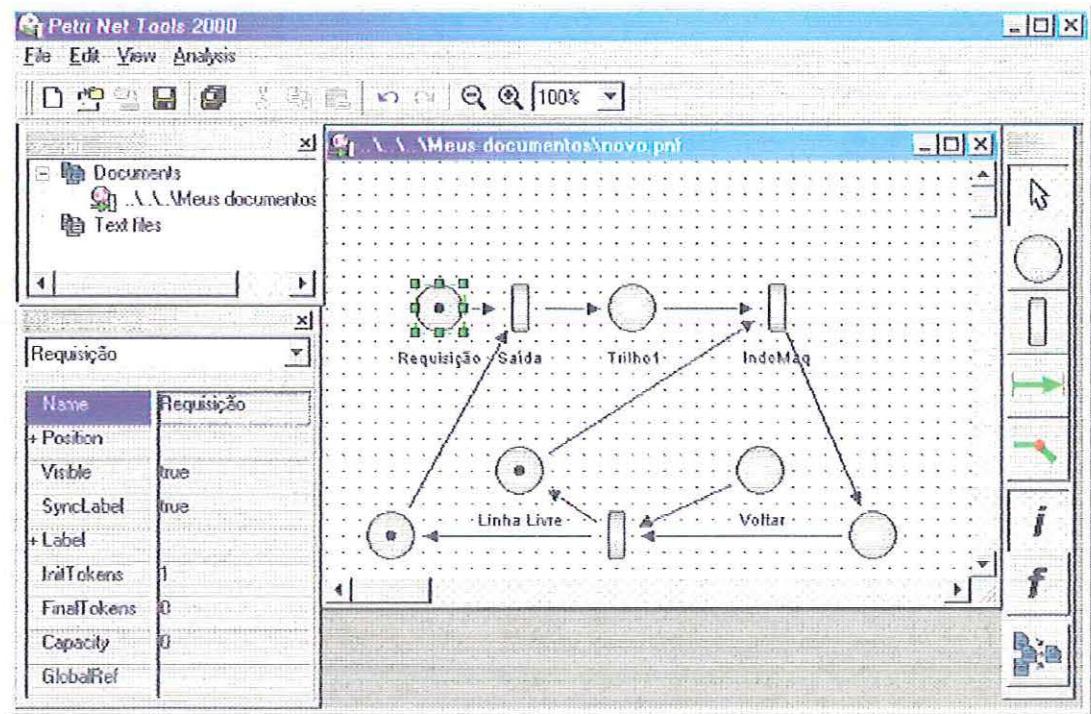


Figura 23. Editor PetriNetTools

O editor permite a manipulação de objetos gráficos na janela de edição (documento). Estão inclusos aqui as funcionalidades de criação, remoção e posicionamento dos objetos; controle e visualização de grade de edição (grid); controle de zoom; cópia e colagem de objeto (cut & paste).

A simulação neste aplicativo auxilia na “validação” do modelo, inclui funcionalidades como a obtenção de árvore de alcançabilidade, a obtenção de matriz de incidência e também auxilia nas análises de propriedades da rede como segurança, limitabilidade e vivacidade. Validar a rede significa chegar à conclusão de que não há erros conceituais a respeito do modelo. Por exemplo: se o número de marcas em um lugar for sempre menor que o peso de seu arco subsequente, a transição associada a este arco nunca será disparada, significando um erro na modelagem. Assim, no editor defini as características de cada lugar, transição e arco do modelo. O resultado é armazenado em um arquivo com dados da rede editada. (PETRI Net Tools, 2001).

#### 4.3.1.1 Interpretação do sistema de armazenamento do Editor

Deve ser possível identificar o formato do arquivo gerado pelo Editor de Redes de Petri para a recuperação dos dados para serem utilizados no

metamodelo. Arquivos das redes desenhadas e “validadas” podem estar no formato texto ou em outro formato, decodificando-os, encontram-se as características do modelo. Descreve a RP=(P,T,F,W,M0) com os dados dos lugares, das transições, dos arcos e das marcas.

A partir deste arquivo, pode-se gerar a RP necessária para executar a simulação desvinculada do editor de RP. Então, deve ser desenvolvido no MCC um interpretador (filtros) dos arquivos dos aplicativos de RP para que se possa executar a simulação com base na leitura de arquivos.

#### **4.3.2 Criação das Classes dos Objetos Virtuais**

As classes criadas para o metamodelo foram voltadas para sistemas de manufatura, mais especificamente para a criação e a simulação de ambientes virtuais sobre rotas de AGVs (Automated Guide Vehicle - veículos autoguiados). O AGV é um integrante importante do sistema de manufatura, que se coloca na posição de fornecer ao sistema, grande flexibilidade de programação MORANDIN (1993), no entanto, a complexidade do projeto e a determinação do seu funcionamento no sistema devem ser consideradas cuidadosamente, no sentido de se aproveitar tal flexibilidade.

Algumas funções e propriedades dos AGVs foram descritas por MORANDIN (1994), INAMASU (1995), PORTO(1993), AUTO SIMULATIONS(1998) e permitiram formalizar e desenvolver os modelos para a simulação das rotas e da criação do AGV virtual.

Foram identificados os elementos dos Sistemas de Manufatura que serão utilizados na simulação e como cada elemento se relaciona com os outros. Como a finalidade de verificar a influência deste comportamento em um ambiente virtual, se fez uma classificação, determinando as classes dos elementos. Assim todos os objetos do sistema serão instância das classes previamente definidas.

Cada classe possuirá duas representações:

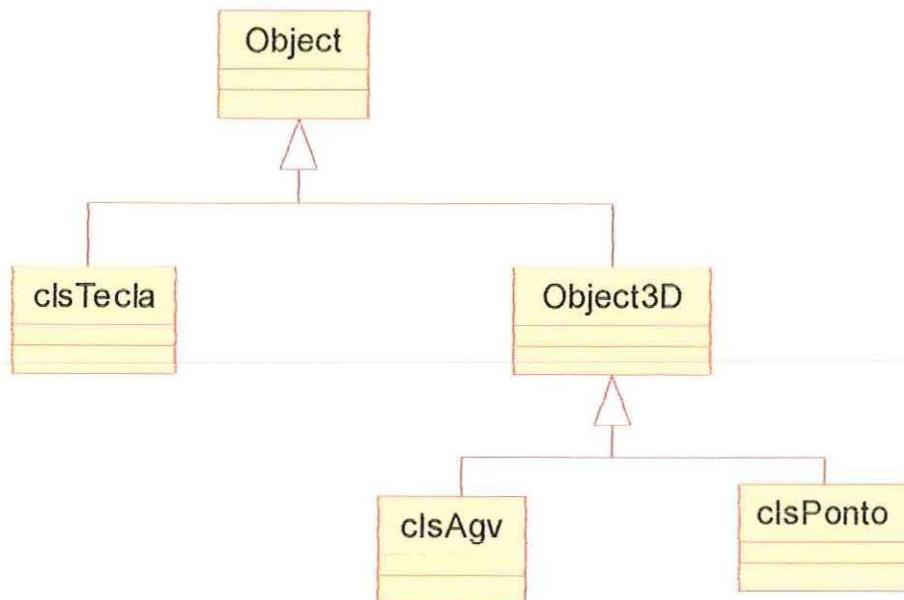
- Uma em linguagem de programação que descreverá as características das classes (atributos) e sua funcionalidade (métodos) com a respectiva implementação.

- Um arquivo de configuração da classe que servirá como guia para o Editor de Ambientes Virtuais. A configuração dá os subsídios para identificar quais são os dados que devem ser requisitados ao usuário e, pela interface do editor, portanto o usuário poderá criar um objeto. O arquivo de configuração também deverá ser utilizado pelo módulo de interligação para poder relacionar os elementos do ambiente virtual com os elementos do modelo de RP. O arquivo de configuração tem um formato determinado.

Inicialmente foram implementadas as classes do AGV, do ponto de controle e das teclas de iteração (Figura 24). O AGV e o ponto de Controle são objetos que vão participar do ambiente e a tecla irá ser um dispositivo de iteração.

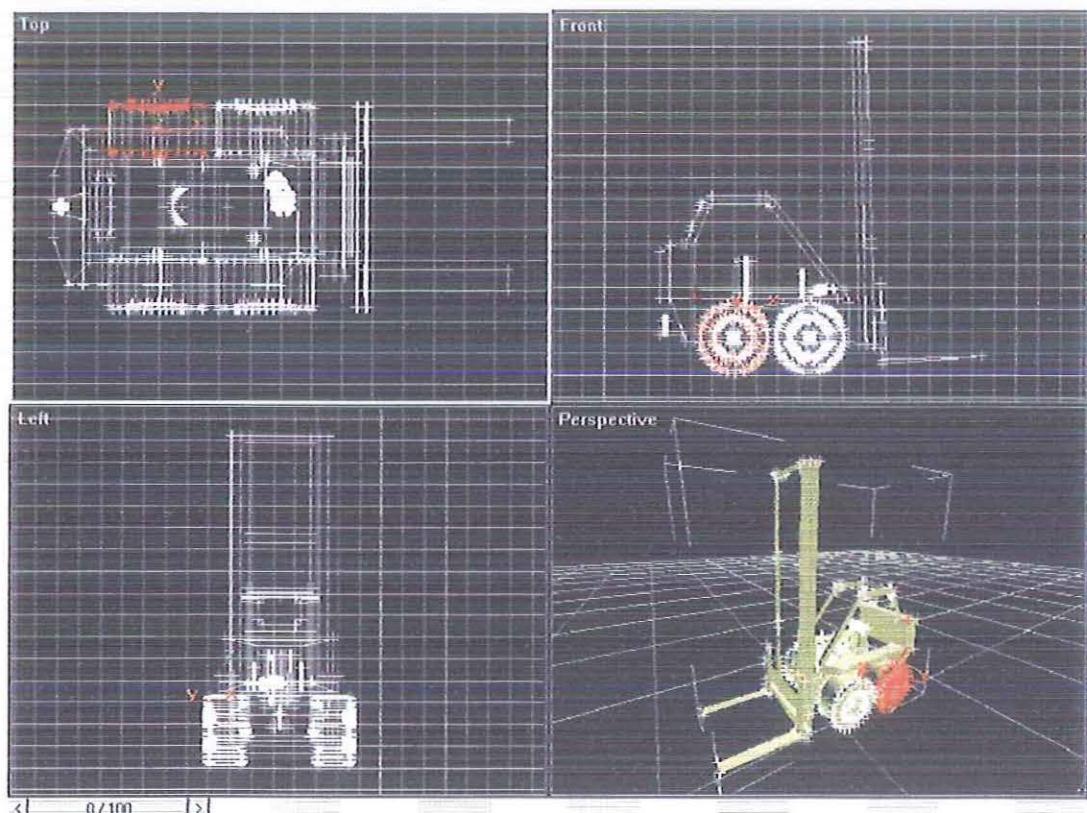
Todas as classes dos objetos virtuais herdarão de uma classe base (classe “Object” da Figura 24) a propriedade nome, e alguns métodos: para alterar as propriedades, reconhecer e executar o método requisitado, atribuir um valor a uma propriedade, pegar o valor de uma propriedade e verificar o estado e efetivar a dinâmica do objeto (*Refresh*). A classe “Object3D” da Figura 24 herda todas as características da classe base (classe “Object” da Figura 24), possui um objeto 3D associado, algumas características comuns aos objetos 3D, tais como os atributos de localização (X, Y, Z) e de direção (direcXZ) além dos métodos de transformações geométricas (Transladar, Rotacionar).

Todas as classes a serem criadas em uma aplicação de manufatura devem necessariamente herdar da classe “Object” e/ou “Object3D”, como por exemplo, a classe de AGV (classe “clsAGV” da Figura 24). A classe AGV possui as características do estado de carga (se está carregado ou descarregado) e a funcionalidade do AGV descrita pelo método “MoverPara”.



**Figura 24.** Classes dos Objetos Virtuais

O modelo 3D armazenado no local especificado no atributo arquivo3D da classe é um elemento geométrico que pode ser desenvolvido e importado de um modelador geométrico (ex: AutoCad, 3Dstudio Max, WorldUP-Modeler) conforme a Figura 25.



**Figura 25.** Desenvolvimento do modelo geométrico 3D

O analista deve informar ao Editor de Ambientes Virtuais (EAV) somente as características que devem possuir valores emanados pelos usuários. O EAV precisa das características para saber os valores iniciais do objeto, os respectivos valores ao criar um objeto. Valores e informações necessárias para a criação de um OV. O usuário também vai ter que relacionar os OVs com o modelo, assim o MI também precisará da estrutura (somente a parte relevante para estabelecer à ligação) da classe.

Portanto, o analista após descrever as classes deve elaborar o arquivo de configuração (Figura 26) que tem uma sintaxe pré-definida, e alguns caracteres são utilizados para informar as características dos dados da classe para o EAV e para o MI:

**nome da classe** - o dado serve para identificar qual a classe do objeto, cada objeto criado no EAV vai ser instanciado no AV, assim o nome da classe serve para o SAV saber que o objeto vai ser instância do “**nome da classe**”.

**/** - um atributo, este caractere serve para informar ao EAV que o usuário pode atribuir um valor ao atributo na criação do objeto. Para saber o tipo do atributo há antes do nome de cada atributo uma letra, tal que i significa que o atributo é do tipo inteiro, e é do tipo enumerado, f é do tipo real, s é do tipo string.

**\*** - um método, serve para informar ao MI que este método pode ser associado com alguma transição do modelo.

**-** - o dado possui apenas importância na estruturação dos atributos dos objetos, o que impõe aos atributos uma hierarquia.

**!** - o dado pode ser associado, informa ao MI que é um atributo que pode ser associado com algum lugar do modelo.

Agv /eEstadoCarga /fX /fY /fZ /sNome *MoverPara(PC, -Estado da Carga --!Carregado --!Descarregado	Agv é Nome da classe EstadoCarga é um atributo tipo enumerado X é um atributo do tipo real Y é um atributo do tipo real Z é um atributo do tipo real Nome é um atributo do tipo string MoverPara(Ponto,) é um método Estado da Carga é um atributo Carregado é um valor do atributo acima Descarregado é um valor do atributo acima
--	--

**Figura 26.** Arquivo de configuração

As classes desenvolvidas para este projeto devem possuir métodos voltados para a comunicação com o sistema de controle. No exemplo do AGV, o método "MoverPara" é ativado por um evento provindo do sistema de controle. Quando o método do objeto virtual finaliza a execução, o método envia uma mensagem ao sistema de controle informando o término da operação.

#### 4.3.3 Editor de Ambientes Virtuais

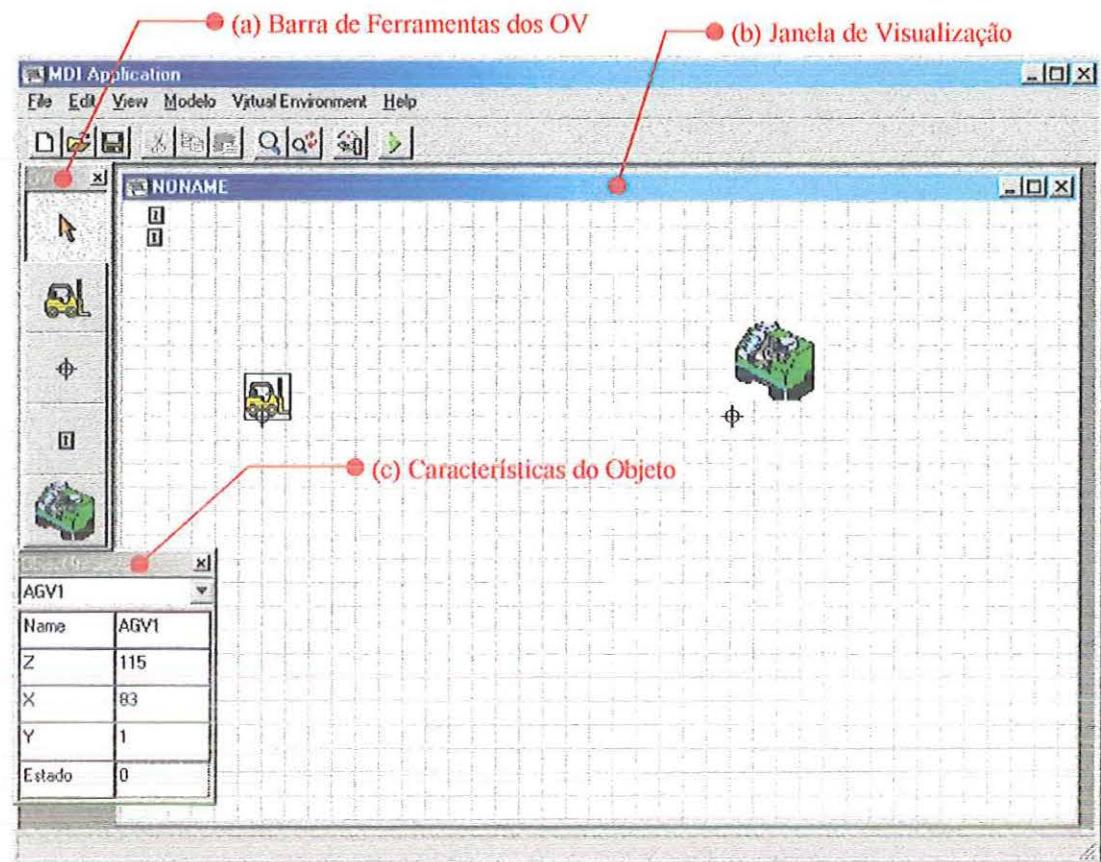
O Editor foi desenvolvido como uma ferramenta flexível para criar a configuração inicial de ambientes virtuais (Figura 27). Ele permite ao usuário inserir no ambiente qualquer objeto derivado das classes da BCOV.

O Editor permite atribuir valores aos atributos do objeto que devem ser configurados pelo usuário do sistema. Portanto, o Ambiente Virtual será composto por objetos definidos neste editor com valores definidos pelo usuário.

O editor proporciona a configuração de AVs de manufatura especificados pelo usuário tal que, elementos de manufatura podem estar em botões na barra de ferramentas, menus para manipulação dos objetos da cena e em caixa de diálogos para auxiliar na entrada de dados dos objetos virtuais.

Os objetos são representados por figuras 2D e são inseridos através da seleção de um botão da barra de ferramenta (Figura 27(a)) e em seguida a iteração (um *click* do mouse) na janela de visualização (Figura 27(b)).

Além da inserção da representação visual o usuário deve preencher as características de cada objeto (Figura 27(c)). Os objetos definidos neste editor são os AGVs, pontos de controle e tecla.



**Figura 27.** Editor de Ambientes Virtuais

O Editor deve produzir os dados dos elementos que vão compor o AV. Ele armazena os dados em um arquivo.

O arquivo armazena dados de cada objeto, de maneira que cada linha possui um objeto. A sintaxe de armazenamento pela notação BNF (Backus Normal Forma) de cada objeto é:

objeto ::= <nome classe>(<parâmetros>)

nome classe ::= AGV|PONTO|TECLA|TORNO

parâmetros ::= <um parâmetro><parâmetros>|<um parâmetro>

um parâmetro ::= <Nome do atributo> '=' <valor atributo>;

O Editor de Ambiente Virtual é uma interface que permite chamar o editor de Redes de Petri Petri Net Tools (2001) e o Módulo de Interligação para elaborar o Ambiente Virtual de Simulação.

O editor foi construído com o paradigma O.O. e ele é constituído basicamente de elementos de interface, assim possui uma classe principal que corresponde ao próprio aplicativo que agrupa classes relacionadas com a edição dos AVs, como:

- A classe da janela do *grid* – local de descrição visual do ambiente,
- As classes dos objetos virtuais – descrição particularizada de cada objeto virtual. A particularização ocorre com a herança, tal que cada objeto pertence somente a uma classe filho e
- a classe do objeto *inspector* – a classe de interface para o usuário inserir os dados.

#### **4.3.4 Módulo de Interligação – MI**

O módulo de interligação foi implementado dentro do servidor de simulação para estabelecer a relação da RP com o AV. A implementação deve habilitar a seleção dos arquivos do AV e do modelo que se deseja ligar, bem como o nome do arquivo que se deseja guardar a Tabela de Conexão. As configurações das classes são necessárias para identificar os elementos que compõem os objetos virtuais (métodos, atributos e valores de atributos) para fazer a relação (Figura 28).



Figura 28. Servidor de Simulação - Entrada de Dados

Após a seleção dos dados necessários para a interligação, faz as associações, sendo que para isto foi desenvolvida uma interface para ligar os objetos virtuais com o modelo de RP, os métodos com as transições e os atributos e valores de atributos com os lugares (Figura 29).

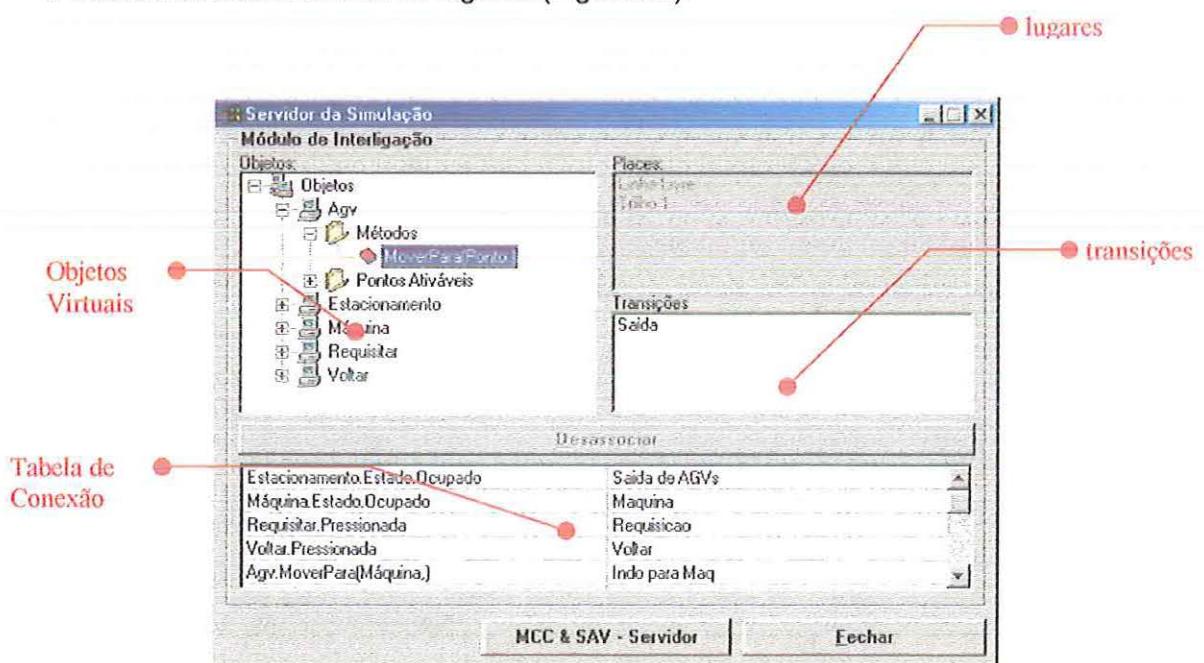
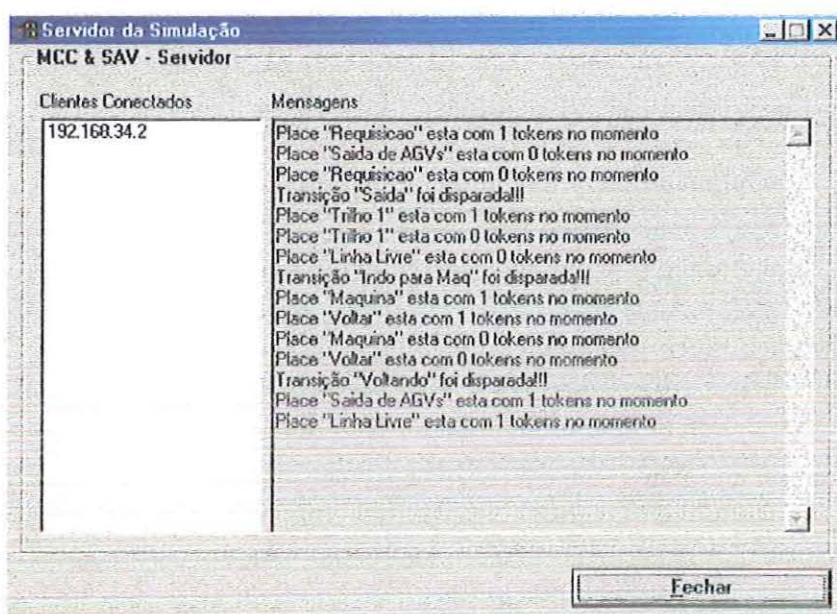


Figura 29. Servidor de Simulação - Módulo de Interligação: Associação

#### 4.3.5 Simulação com Interface de RV

Foi implementado um sistema servidor de simulação, o sistema inicialmente requer: nome do arquivo da RP, o AV e uma Tabela de Conexão (pode selecionar uma TC existente ou pode chamar o MI para criá-la). (Figura 28)

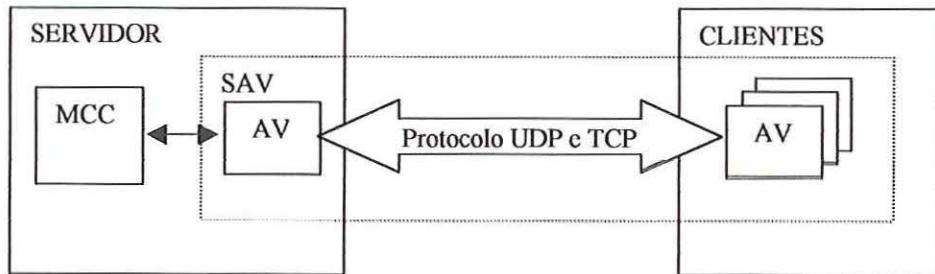
Esses arquivos são fontes de dados para criar a RP em memória e criar o AV. Na seqüência, fazer a comunicação de ambos, ou seja, os dados do arquivo são necessários para executar a simulação (Figura 30).



**Figura 30.** Servidor de Simulação – Clientes Conectados e Início da Execução da Simulação

Definiu que nesta implementação se deixaria o processamento lógico no servidor e a apresentação audiovisual no cliente. No servidor estará o Módulo Central de Controle e uma parte do Simulador de Ambientes Virtuais (SAV – servidor). (Figura 31)





**Figura 31.** Comunicação do Servidor/Cliente

Nesta implementação o servidor pode possuir vários AVs interligados, sendo que cada AV pode possuir capacidades de processamentos diferentes e por este motivo é necessária a preocupação com a sincronia dos clientes. Para solucionar este problema optou-se pela centralização do AV, ou seja, um AV é tratado em apenas uma máquina (servidor), com isto os clientes apenas se dedicam na apresentação audiovisual. No caso de uma apresentação ser mais lenta que a freqüência com que se recebe novo pacote, o sistema cliente irá descartar os pacotes mais antigos, mantendo assim a sincronização dos AVs.

O SAV – servidor foi implementado como sendo uma única instância responsável em fazer as atualizações das coordenadas e dos estados do AV e a outra parte foi implementada em outro software, SAV – cliente, que pode possuir várias instâncias em máquinas diferentes, de tal maneira que cada instância é utilizada apenas como uma representação audiovisual do AV que também é responsável por capturar e retransmitir as iterações dos dispositivos de entradas utilizados pelos usuários deste software ao SAV - servidor.

Utilizou-se protocolo UDP (User Datagram Protocol) com o tratamento de erro feito pela própria aplicação nos casos das mensagens de iterações vindas dos terminais, utilizando retransmissão de dados para confirmação.

A diferença dos protocolos UDP e o TCP (Transmission Control Protocol) é que no primeiro os serviços não são orientados à conexão e no TCP os serviços são orientados à conexão (TCP).

Um serviço que é orientado à conexão pode ser considerado como o sistema telefônico que possui três fases: estabelecimento da conexão, transferência de dados e término da conexão, aonde a ordem da transmissão dos dados é preservada. Já um serviço não orientado a conexão pode ser comparado com o

sistema de correios, onde as mensagens possuem o endereço do destinatário e o número da seqüência, além de serem roteadas independentemente das outras.

Porém existe também um outro conceito de rede que diferencia um protocolo do outro, que é a confiabilidade. Os protocolos ditos confiáveis (TCP) garantem a entrega dos dados e também possuem um mecanismo de confirmação do recebimento da mensagem com retransmissão. Os não confiáveis (UDP) não garantem a entrega dos dados e não possuem um mecanismo de reconhecimento das mensagens.

O protocolo TCP é orientado à conexão e é confiável, mas na comunicação entre as duas máquinas o UDP seria mais rápido, porém com a possibilidade de ter dados errôneos na máquina de destino. Por esta razão o protocolo UDP não é um protocolo indicado para aplicações com transmissão de arquivos, porém indicado para a transmissão de vídeo ou som onde o recebimento ou não com erro de uma parte da mensagem não implicaria em prejuízo para a aplicação destino e além disso, essas aplicações necessitam de altas velocidades de transmissão.

Atualmente existem muitos estudos sobre sistemas de realidade virtuais distribuídos e a comunicação na rede (ENCARNAÇÃO&FRAUNHOFER, 1998) e (ARAÚJO, 1996).

ZHOU-YI, et al. (2000a) e ZHOU-YI, et al. (2000b) estudaram e analisaram as tecnologias TCP/IP e UDP para a comunicação entre clientes e servidor de ambientes virtuais. E certificaram que a UDP tem menor tempo de latência e a TCP/IP tem como vantagem a consistência dos dados recebidos e transmitidos, os autores estão estudando um protocolo de comunicação que possa conciliar velocidade de reação com a consistência dos dados.

#### **4.3.5.1 Simulador de Ambientes Virtuais**

O Simulador de Ambientes Virtuais foi dividido em duas partes: o SAV – servidor é responsável em efetuar os processos de transformação do objeto e os SAVs - clientes são responsáveis em renderizar as cenas do ambiente (traçar a imagem no dispositivo de vídeo) e em captar as interações dos usuários e transmiti-las para o SAV -servidor. Assim:

- O SAV - servidor é um sistema que possui uma coleção de referências às classes do BCOV, assim os objetos virtuais são instâncias dinamicamente de acordo com os dados definidos no EAV. As operações existentes no SAV - servidor servem para efetuar a comunicação do MCC com os objetos virtuais e vice-versa estabelecendo dinâmica das instâncias das classes do BCOV;
- O SAV - cliente é um sistema que trabalha como um terminal de entrada e saída de dados, mas não é responsável pelo processamento das informações (exceto a navegação do ambiente, iluminação), apenas traduz as informações digitais (recebidos via rede do SAV - servidor) em informações audiovisuais nos dispositivos de saídas e capta as iterações dos periféricos utilizados pelos usuários e envia para o SAV – servidor para o tratamento da mesma. Há ferramentas para o auxílio de desenvolvimento de objetos virtuais e Ambientes Virtuais, como a biblioteca WorldToolKit da SENSE 8, a qual foi utilizada neste projeto para a criação de Objetos Virtuais e neste tópico para construir o ambiente virtual.

O SAV é uma ferramenta que pode ser chamada por um comando embutido na interface do Editor de Ambientes Virtuais ou por linhas de comandos ou por ícone no ambiente de trabalho. O usuário tem que escolher um ambiente já editado no Editor de Ambientes Virtuais e um modelo já editado no MEVM.

O SAV - cliente pode ser carregado na própria máquina do SAV – servidor ou em outra máquina. Caso exista uma simulação em andamento, o usuário tem a possibilidade de conectar-se à mesma obtendo o AV em seu estágio corrente.

#### **4.3.5.2 Sistema de Controle**

O Módulo Central de Controle (MCC) irá fazer o processamento das informações, solucionando os problemas com base no processo que intervém entre a entrada e saída, buscando os objetivos subsequentes de um estado inicial e/ou de uma iteração do usuário.

O Módulo Central de Controle - MCC será responsável em colocar a RP na memória e processá-la para gerenciar o AV. Assim o primeiro passo é interpretar o arquivo criado no Editor de Redes de Petri e colocá-lo na memória.

O MCC deve possuir um filtro para identificar o formato do arquivo em RP. Assim, o primeiro passo é identificar a forma do arquivo gerado pelo editor. No caso desta implementação foi utilizado e identificado o arquivo gerado pelo editor que antecedeu o Petri Net Tools 2001. A Figura 32 é um exemplo do arquivo, analisando os dados foi possível identificar como o MCC deverá recuperá-los para a sua utilização.

(1)	(483,164)	(0,0)	(0,0)
Petri Net graphic file	-21	0	0
10206	6		
6	0	(0,10)	(0,10)
3	0		
10	0	6	15
26		3	8
(103,216)	0	1	1
-0	Voltar		
1	(0,25)	2	14
1	(252,51).....(2)	3	3
0	1	(0,0)	(0,0)
0	1	(0,0)	(0,0)
0	0	0	0
0	0		
Saida AGVs	0	(0,10)	(0,10)
(0,25)			
(365,51)	1	8	17
2	Saida	4	9
2	(0,25)	1	1
1	(498,51)		
0	3	3	9
0	2	7	14
0	0	(0,0)	(0,0)
0	0	(0,0)	(0,0)
Trilho 1	0	0	0
(0,25)			
(609,216)	1	(0,10)	(0,10)
7	Indo para Maq		
3	(0,25)	10	23
0	(351,216)	5	10
0	9	1	1
0	3		
0	0	7	21
0	0	9	9
Maquina	0	(0,0)	(0,0)
(0,25)		(0,0)	(0,0)
(131,51)	1	0	0
12	Voltando		
4	(0,25)	(-0,10)	
0	4.....(3)		(0,10)
0	1	11	
0	1	6	
0		1	
0	0		
Requisicao	1	9	
(0,25)	(0,0)	0	
{265,139}	(0,0)	(0,0)	
14	0	(0,0)	
5		0	
1	(0,10)	1	
0		(0,10)	
0	5		
0	2	13	
0	1	7	
Linha Livre		1	
(0,25)	1		
	2	12	
	(0,0)	1	
		(0,0)	

**Figura 32.** Arquivo gerado por um editor de Redes de Petri

O arquivo da Figura 32 apresenta diversos valores que somente têm sentido para o editor de redes de petri, como por exemplo valores para situar na tela os objetos desenhados. O primeiro valor de relevância é o número localizado na 3<sup>a</sup> linha da primeira coluna (neste exemplo o número é 6) que representa a quantidade

de lugares, seguido pela quantidade de transições e a quantidade de arcos do modelo:

Petri Net graphic file  
10206  
6 o número de lugares do sistema  
3 número de transições do sistema  
10  
26 o número de arcos

Características dos lugares.:

(103,216) valores que somente têm sentido para o editor de redes de petri  
0 valor exclusivo para cada elemento do modelo  
1 identificador do lugar  
1 o número inicial de marcas  
0 o número final  
0 capacidade do lugar  
espaço  
0 0 indica que este elemento é um lugar  
Saída AGVs o nome do lugar  
(0,25) valores que somente têm sentido para o editor de redes de petri  
Isto se repete para todos os *lugares*

As características das transições,

(252,51) valores que somente têm sentido para o editor de redes de petri  
1 valor exclusivo para cada elemento do modelo  
1 identificador da transição  
0 }  
0 estes números formam o tempo de disparo no formato x,xx  
0 espaço  
1 1 indica que este elemento é uma transição  
Saída o nome da transição  
(0,25) valores que somente têm sentido para o editor de redes de petri  
Isto se repete para todas as transições

Características dos arcos:

4 valor exclusivo para cada elemento do modelo  
1 a identificação do arco  
1 o peso deste arco  
0 espaço em branco  
0 origem do arco  
1 depois o destino

Após a leitura e colocação da RP na memória o MCC deve ficar ativo conforme o Diagrama de Estado da Figura 20, e ele deve executar as suas atividades representadas na Figura 16. O MCC processa a RP conforme o formalismo da própria ferramenta, portanto quando ocorre conflito é eleita aleatoriamente uma transição para ser disparada. A Visão Lógica do Sistema de controle implementado é mostrada no Apêndice B.

#### **4.4 Desenvolvimento de um sistema de simulação de Manufatura com interface de RV**

O sistema computacional implementado a partir do metamodelo é uma ferramenta de desenvolvimento de sistemas de simulações de manufatura com interface de RV. Mas, o uso ou aplicação da Simulação requer mais do que os conhecimentos de como se usar um software de simulação.

Na realidade, ele deve ser caracterizado como um estudo de simulação, que por natureza, é um projeto e, como em qualquer projeto, haverão tarefas para serem realizadas e recursos humanos e materiais que serão necessários.

Para um projeto de simulação ser bem sucedido, ele deve ser planejado com o entendimento das exigências de cada uma das tarefas envolvidas.

As falhas normalmente são resultados de atitudes precipitadas, tomadas durante uma simulação, sem primeiro considerar os passos envolvidos e sem desenvolver um plano para o procedimento.

Os passos a seguir são recomendados para o usuário do sistema computacional implementado. O usuário para criar e executar a simulação do sistema desejado, deve fazer a:

1. Concepção ou formulação do modelo
2. Implementação e validação do modelo
3. Edição da Interface gráfica para o modelo
4. Integração do modelo com AV
5. Execução

Apesar destes passos estarem dispostos em uma certa seqüência linear, isto não precisa ocorrer exatamente desta maneira em um estudo prático de simulação, pode haver várias iterações e realimentação no processo à medida que o entendimento do problema muda.

#### **4.4.1 Concepção ou formulação do modelo**

Na primeira etapa cria-se o modelo do sistema de manufatura que se deseja simular. O analista de simulação deve entender claramente o sistema a ser simulado e os seus objetivos através da discussão do problema com especialistas. Deve-se decidir qual é a abrangência do modelo e o nível de detalhe. Todas as hipóteses devem ser claramente estabelecidas. (CHWIF, 1999). Após estas decisões, o modelo que está na mente do analista (modelo abstrato) deve ser representado em Redes de Petri, técnica de representação de modelo, a fim de torná-lo um modelo conceitual.

#### **4.4.2 Implementação e validação do modelo**

Na segunda etapa, o modelo conceitual deve ser editado no computador no MEVM, editor de Redes de Petri, para colocar o modelo em formato adequado para um sistema computacional. O modelo computacional deve, ainda ser verificado, a fim de avaliar se está operando de acordo com o pretendido. Alguns resultados devem ser gerados para validar o modelo, observando se o modelo é uma representação precisa da realidade. O Editor Petri Net Tools 2000 possui algumas ferramentas de análise.

#### **4.4.3 Edição da Interface gráfica para o modelo**

Neste passo deve-se definir a configuração do ambiente virtual no editor. Lembrando que a edição não representa a lógica e nem a dinâmica de execução, e sim a descrição dos objetos pertencentes ao ambiente e seu estado inicial.

Faz-se a edição gráfica através do Editor de AV, se há elementos de manufatura suficientes na biblioteca, não haverá a necessidade de programação, caso contrário, deve-se criar o objeto que se deseja e disponibilizar no editor de AV.

#### **4.4.4 Integração do modelo com AV**

Faz a integração do modelo com o AV, através do módulo de interligação (MI), que vai relacionar os *lugares* e transições do modelo com os atributos e métodos dos Objetos virtuais respectivamente. Este passo reveste o modelo com uma interface em RV.

#### **4.4.5 Execução**

Nesta etapa, após a integração, o sistema está pronto para a realização de experimentos, de maneira que se deve executar o MCC em um computador servidor e os AVs devem ser executados em computadores clientes conectados ao servidor (*host*).

Várias “rodadas” do modelo podem ser efetuadas. Os resultados da simulação são analisados, a partir destes, conclusões e recomendações sobre o sistema podem ser gerados. Caso necessário, se o resultado da simulação não for satisfatório, o modelo pode ser modificado, e estes passos são reiniciados.

### **4.5 Estudo de Caso**

Para exemplificar o sistema implementado foi feita a simulação de ambientes virtuais com interface de RV em um estudo de caso simples sobre rotas de AGVs (veículos autoguiados).

A funcionalidade do sistema será exemplificada com o estudo de caso em cada um dos módulos do metamodelo.

#### **4.5.1 Passo 1 - Criar o modelo**

Este passo é bem complexo e com certeza um dos mais importantes no sucesso da simulação. Um modelo não é verdadeiro nem falso, mas útil ou não útil. Um modelo útil é aquele que é válido e proporciona as informações necessárias para encontrar os objetivos da simulação.

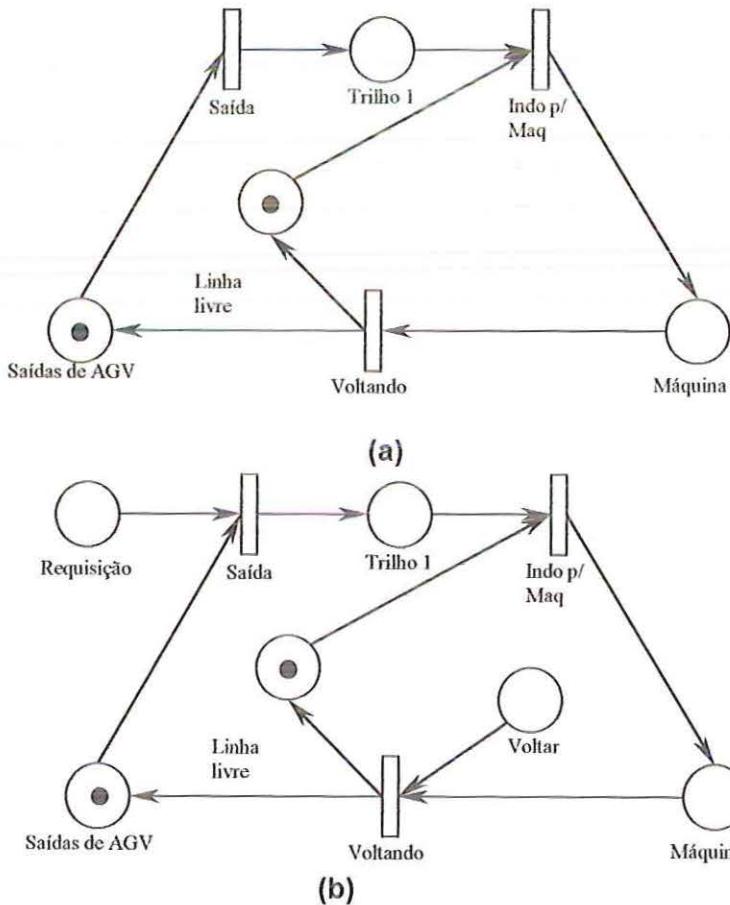
Uma vez que tenham sido coletadas as informações suficientes para definir a operação básica do sistema, a atividade de construção do modelo pode ser iniciada.

O processo de modelagem utilizado neste trabalho representa os caminhos que um AGV pode percorrer saindo do lugar “Saída de AGVs” passando pelos lugares “Trilho 1”, “Máquina” e de volta ao ponto inicial, conforme Figura 33(a). Também é possível visualizar se há ou não AGV no lugar “Máquina” em um dado momento, simbolizado pela marca colocado no lugar “Linha Livre”. Na segunda Figura 33(b), acrescentou-se o lugar “Requisição” para requisitar a saída de algum

veículo, bem como o lugar “Voltar” para chamar de volta o veículo que esteja na “Máquina”.

Para construir uma rede de petri com intuito de possibilitar a iteração do usuário no ambiente a ser simulado deve-se acrescentar as interferências do usuário ao modelo, Figura 33(b). Por exemplo, um AGV somente sai da “Máquina” se receber a ordem para isto, simbolizada pelo lugar “Voltar”, a ordem pode ser dada através de uma iteração do usuário.

Uma transição dispara se tem *marcas* nos *lugares* anteriores ( $\bullet t$ ) e se estas forem em maior número do que o peso dos respectivos arcos. Então, o lugar “Linha Livre” fica como um evento interno ao programa, que serve somente para que a rede seja capaz de resolver se permite que um veículo se desloque para a “Máquina” ou não. A transição “saída” e o lugar “Trilho 1” poderiam ser suprimidos, mas foram deixadas para verificar que nem todos os elementos da RP têm que ter o seu correspondente no AV.



**Figura 33.** Rota dos AGVs modelado em Redes de Petri

Como este estudo de caso tem como intuito verificar o aplicativo construído foi tomado como base a figura 33(b). O lugar “Requisição” pode estar simbolizado no ambiente virtual como um botão de chamada de AGV; o lugar “Voltar” também pode ser um ponto de ativação e até mesmo o lugar “Saída de AGVs” que pode simbolizar um galpão.

#### **4.5.2 Passo 2 – Editar o modelo e validá-lo**

No Editor o usuário pode criar lugares e transições comuns em mais de um arquivo do projeto, denominados lugares e transições globais. Todos os arquivos do projeto podem então ser integrados, formando um único modelo. O modelo do passo 1 deve ser editado no editor de Redes de Petri, a Figura 23 exibe um exemplo de rede de Petri modular editado no Petri Net Tools 2000 sem lugares e transições globais.

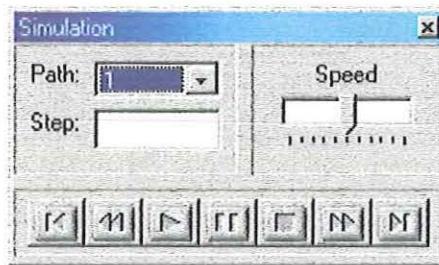
Finalmente, o modelo deve ser analisado. Após a edição deve-se fazer alguns testes. O processo de demonstração que um modelo trabalha como se pretende, é referido como Verificação do Modelo, pois uma vez que um modelo é definido este deve ser executado para assegurar que trabalha corretamente.

Há algoritmos de Redes de Petri que possibilitam verificar as propriedade do modelo e fazer a análise, para ver o quanto o modelo corresponde à definição do sistema real. Caso estes procedimentos sejam executados sem encontrar uma discrepância entre o sistema real e o modelo, este é dito ter aspecto de validade.

No modelo elaborado, o AGV só pode se locomover de um ponto a outro se ele for chamado, caso contrário, ele fica estático onde está. Como num modelo para Ambiente Virtual muitas marcas vão estar associadas a iterações do AV assim, as marcas são colocadas em tempo de execução e não por de uma transição, o sistema de análise vai ser extenso, pois tem que verificar as possíveis iterações do usuário, atribuindo as marcas aos respectivos lugares.

Este editor possui alguns testes, as possíveis formas de análise disponíveis consistem na simulação, identificação de propriedades comportamentais do modelo, geração da matriz de incidência, geração da árvore de alcançabilidade, e análise usando invariantes. Todas essas funcionalidades estão disponíveis no menu “Analysis”.

A simulação (Figura 34) fornece mecanismo ao usuário para escolha do caminho desejado (soluções da árvore de alcançabilidade) e de controle da simulação (passo a passo ou continuamente) do modelo editado no editor (Figura 23).



**Figura 34.** Janela de simulação

As propriedades comportamentais do modelo são simplesmente disponibilizadas através da janela de propriedades do modelo visto na Figura 35(a).

A matriz de incidência e a árvore de alcançabilidade são disponibilizadas em formato texto. As saídas geradas pelo editor podem ser visualizadas na Figura 35(b) e (c) respectivamente.

Por fim, a análise de invariantes, implementada para os S-invariantes, consiste em identificar se a soma de marcas de um conjunto de lugares é constante em qualquer marcação alcançável. Assim o usuário seleciona um conjunto de lugares e o sistema retorna se eles são ou não invariantes, neste modelo todos os lugares não são invariantes.

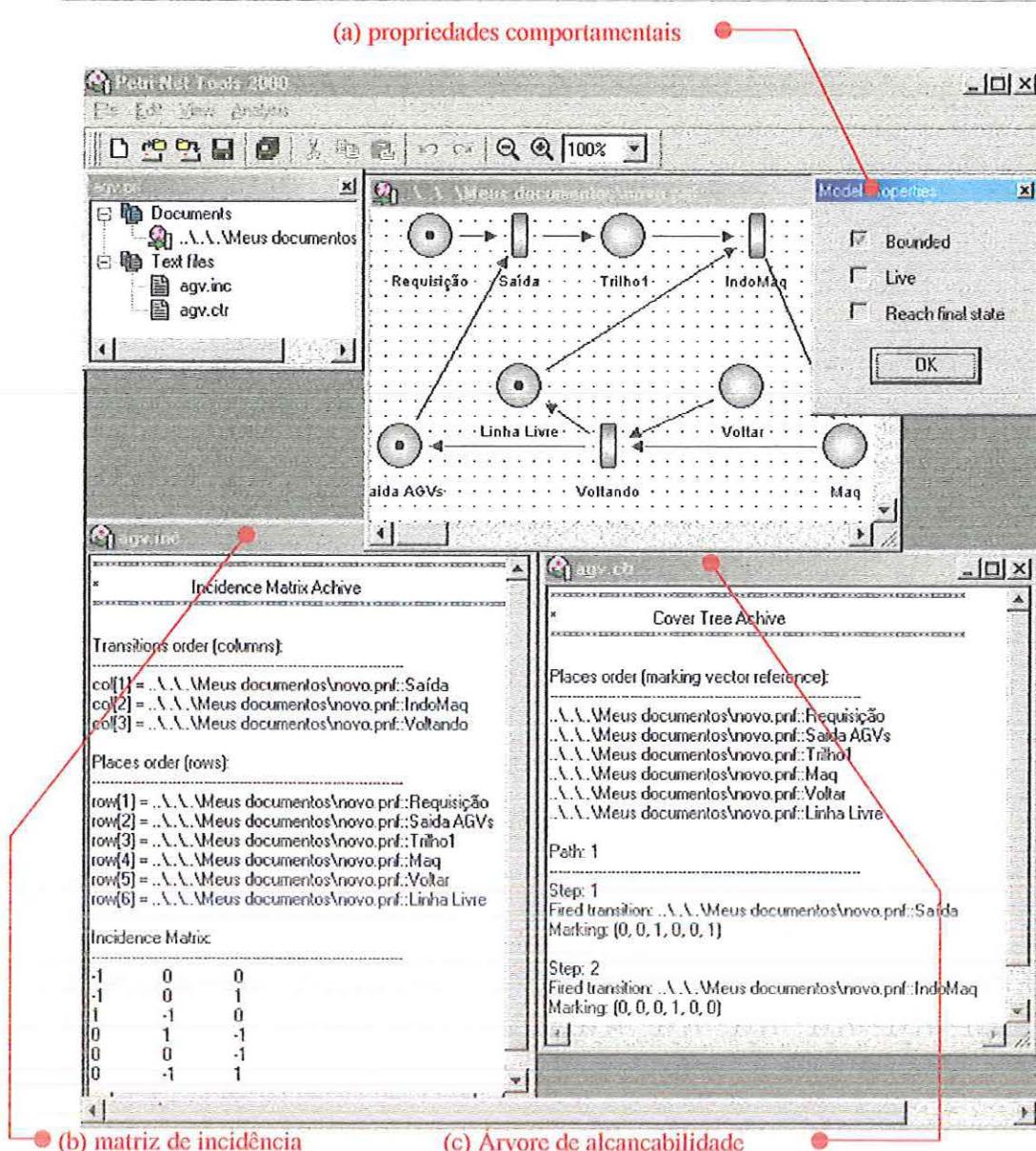


Figura 35. Propriedades e Análise da RP

O próximo passo seria pegar o estado intermediário desta simulação e inserir uma nova marca na iteração “Voltar” e fazer uma nova análise.

Ainda, se o modelo não possui marcas nas iterações do usuário quando inicia a análise, a rede fica estática, ou seja o estado da rede não muda pois não há nenhuma transição habilitada para o disparo e se o modelo possuísse marcas em todos os lugares de iterações fica mais fácil de analisar, em contrapartida pode-se perder algum ponto relevante.

#### **4.5.3 Passo 3 - Edição do Ambiente Virtual**

Neste passo deve-se definir a configuração do ambiente virtual no editor. Lembrando que a edição não representa a lógica e nem a dinâmica de execução, e sim a descrição dos objetos pertencentes ao ambiente e seu estado inicial.

No AV inseriu-se um AGV e dois pontos de controle representados por “Saida AGVs” e “Máquina”, e duas teclas de iteração representando “Requisição” e “Voltar”. Cada objeto inserido no AV foi configurado conforme as informações do usuário. Após a entrada de dados, deve-se salvar o AV em um arquivo. Conforme a Figura 27.

O Editor do Ambiente Virtual vai gerar o arquivo com os dados mostrados na Figura 36.

```
AGV(Nome=AGV1,X=83,Y=1,Z=115,EstadoCarga=0,)  
PONTO(Nome=Estacionamento,X=83,Y=1,Z=115,Estado=1,)  
PONTO(Nome=Máquina,X=200,Y=1,Z=115,Estado=0,)  
TECLA(Nome=Requisitar,Valor=R,)  
TECLA(Nome=Voltar,Valor=V,)  
TORNO(Nome=Torno1,X=210,Y=1,Z=115,-)
```

**Figura 36.** Arquivo do Ambiente Virtual gerado pelo Editor

O Servidor de Simulação cria o AV com os dados deste arquivo (Figura 36). O arquivo possibilita identificar quais os objetos que serão criados com as respectivas configurações iniciais (*setup*) e de qual classe o objeto será instância. Os objetos são criados em tempo de execução pelo simulador.

#### **4.5.4 Passo 4 – Integrar o Modelo com o Ambiente Virtual**

Neste passo devem interligar os dados do modelo editado no Passo 2 com os dados do Ambiente Virtual editado no Passo 3. Assim, deve-se selecionar o arquivo do AV e o arquivo do modelo em RP que se deseja ligar e o nome do arquivo que se deseja guardar a Tabela de Conexão (Figura 28). Precisa-se também das configurações das classes para identificar os elementos que compõe os objetos virtuais (métodos, atributos e valores de atributos) para poder fazer a relação.

As associações devem ligar os métodos com as transições e os atributos e valores com os lugares, conforme a Figura 29.

A interface para interligar o AV e o Controle é uma caixa de diálogo possui quatro quadros (Figura 29). No primeiro quadro tem-se os objetos virtuais com os seus métodos e seus pontos ativáveis (atributo com valores que podem ser associados com os lugares do modelo), no segundo quadro à direita tem-se os lugares do modelo, e o terceiro à direita abaixo tem-se as transições. Assim, escolhe-se um ponto ativável no primeiro e o *lugar* no segundo quadro ou escolhe-se um método no primeiro e uma transição no terceiro quadro, clicando no botão *Associar*, a ligação é estabelecida. Os itens relacionados são apresentados no quadro abaixo. Este módulo será responsável pela criação da tabela de conexão (quarto quadro) para relacionar os nomes dos pontos ativáveis e dos métodos virtuais com os elementos da simulação.

Quando efetua a associação de uma transição com um método que possui parâmetros abre-se uma caixa de diálogo. O usuário deve entrar com os parâmetros na caixa de diálogo, se o parâmetro é de um tipo previamente conhecido, por exemplo: se o tipo é numérico então deve-se editar o número; se o tipo é um objeto então deve-se mostrar todos os objetos deste tipo definido no EAV, Figura 37.



Figura 37. Módulo de interligação – Parâmetros dos Métodos

A tabela de conexão é guardada em um arquivo para ser utilizado na execução da simulação, o arquivo gerado corresponde à Figura 38.

```
Requisitar.Pressionada#Requisicao
Voltar.Pressionada#Voltar
Estacionamento.Estoado.Ocupado#Saida de AGVs
Máquina.Estoado.Ocupado#Maquina
Agv.MoverPara(Máquina,)#Indo para Maq
Agv.MoverPara(Estacionamento,)#Voltando
```

Figura 38. Tabela de conexão gerado pelo MI

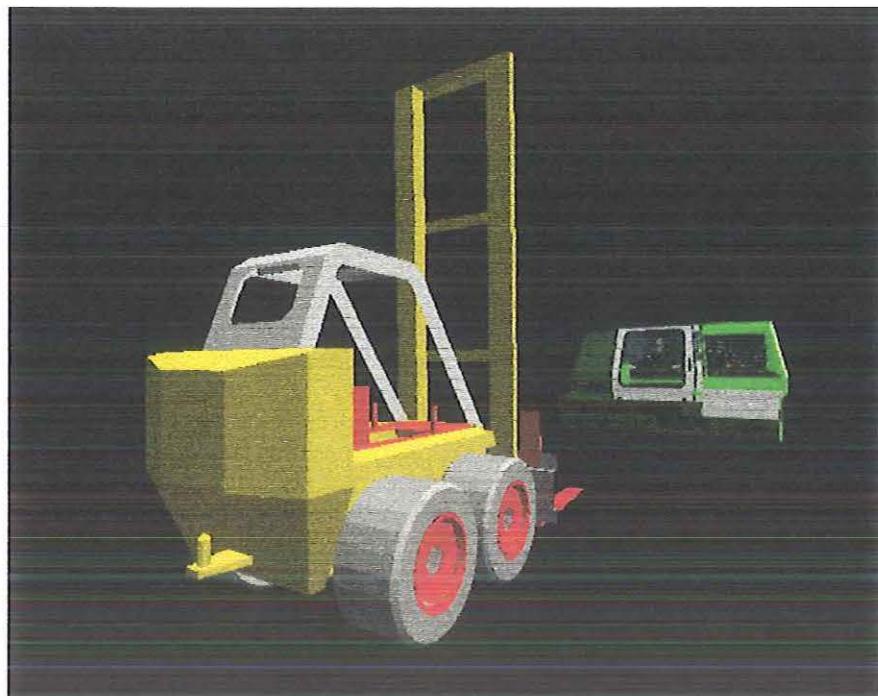
#### **4.5.5 Passo 5 – Executar a Simulação**

Este passo deve conduzir o experimento de simulação, ele é dividido em dois sistemas: o servidor e o cliente.

O usuário do sistema servidor vai selecionar o projeto a ser executado (RP, AV eTC) conforme a Figura 28. O servidor fica esperando os clientes se interligarem.

Os usuários dos sistemas clientes devem fornecer o IP da máquina do servidor que se deseja conectar. O cliente ao conectar-se pode acompanhar e participar da simulação de um Ambiente Virtual, quando a simulação for iniciada ou instantaneamente.

Quando o usuário do servidor inicia a simulação, automaticamente os sistemas clientes abrem uma janela com o Ambiente Virtual (Figura 39). O usuário cliente tem o poder de navegar em seu ambiente através de dispositivos de iteração como *mouse*, teclado ou HMD (neste protótipo o único meio de navegação é através do *mouse*). A iteração de navegação não altera o estado do ambiente ela simplesmente muda o ponto de vista (*view point*). Quando ocorre uma iteração que pode repercutir em uma reação do Ambiente (solicitação de mudança no estado do ambiente) é enviado ao servidor para ser tratado.



**Figura 39.** Ambiente Virtual

A composição do ambiente virtual acima foi configurada no Editor de Ambiente Virtual (passo 3), com: um AGV, um Torno (ambos podem ser visualizados no AV), dois pontos de controle, e duas teclas de iterações “R” e “V”. O controle é efetuado através da rede de Petri. O AV e o controle se entendem devido a Tabela Conexão. Assim, quando a tecla “R” é apertada o AV envia esta informação para o controle e ele faz o tratamento, caso o AGV esteja no Estacionamento e a Linha esteja Livre o sistema de controle vai deixar a transição Indo para a Máquina ativa .

#### **4.5.6 Observações**

A visibilidade do comportamento e dos elementos em cena dependerá do usuário do sistema computacional. No estudo de caso pode-se reparar que no modelo tem lugares (“Trilho1” e “Linha Livre”) e uma transição (“Saída”) que estão presentes, mas estes elementos não foram associados com o Ambiente Virtual, e no ambiente virtual criou-se um objeto que não é associado ao modelo (“Torno”) mas pode visualizá-lo no ambiente (Figura 39).

## 5. Conclusões

Atualmente, um dos principais ápices da tecnologia da informação é a Realidade Virtual (RV), mas a complexidade envolvida na construção do ambiente virtual distânciaria esta tecnologia da simulação de Sistemas de Eventos Discretos (SED). A aproximação de ambas foi proposta por aplicativos baseados no metamodelo e verificou-se que:

- Nos estudos realizados e apresentados foi demonstrada a viabilidade de construir simuladores nos quais os usuários finais podem desenvolver simulações a SED com interface de RV sem a necessidade de codificação computacional e sem ter que possuir grandes conhecimentos no desenvolvimento de sistemas de RV. Assim, pode-se afirmar que os objetivos propostos para este trabalho foram alcançados, facilitando o desenvolvimento de softwares para as simulações com Interface em RV.
- No trabalho a aplicação de um método formal como a RP para o controle dos Avs supriu a atual carência de formalismo no desenvolvimento de Avs.
- A ferramenta proporciona o desenvolvimento de simulações com AV distribuídos, assim várias pessoas separadas fisicamente podem atuar em um mesmo ambiente, elas vêem as repercuções das suas ações e das outras que estão conectadas no mesmo ambiente.
- E, finalmente, a representação da RP sob a forma de um grafo e a representação das classes dos objetos virtuais (OV) das bibliotecas sob o paradigma de Orientado a Objetos possui um grande poder de expressão, simplificando a comunicação entre o projetista e o sistema de simulação.

Esta comunicação permite ao projetista desenvolver a simulação editando apenas o modelo graficamente através do editor de RP e criando o AV com a inserção dos OV por meio do Editores gráfico de Ambientes Virtuais.

Como limitações, tem-se a complexidade envolvida na programação das classes da Biblioteca de Classes de Objetos Virtuais (BCOV), assim quando a biblioteca não possui a classe do objeto desejado deve-se desenvolver o respectivo código. Quanto maior o ambiente e mais complexo maior será o seu modelo, o que dificulta o entendimento e o relacionamento da RP com os objetos virtuais.

## 5.1 Trabalhos Futuros

A utilização do paradigma orientado a objetos na implementação do Módulo Central de Controle (MCC) proporciona a capacidade de se definir extensões de RP pelo conceito de herança de forma que as classes atuais sirvam como classes bases, e consequentemente a RP poderia realizar o tratamento de problemas analíticos e estatísticos, características presentes em sistemas discretos.

Durante a realização do trabalho foi verificado que a qualidade dos sistemas de simulação com RV vai estar atrelada a:

- ao modelo desenvolvido e a sua validação,
- a qualidade gráfica,
- as iterações e
- as imersões.

A qualidade e representatividade do modelo dependem da experiência do usuário. A modelagem requer boas habilidades analíticas, estatísticas, de comunicação, em organização e em engenharia. O modelador deve entender do sistema investigado e ser hábil para escolher através do complexo relacionamento de causa e efeito, o que determinará a execução do sistema. Quando em comunicação com os requisitantes e usuários durante um estudo de simulação, também será vital assegurar que o modelo proposto seja utilizado e que todos entendam os objetivos, suposições e resultados do estudo.

---

No protótipo desenvolvido para que ele adquirira maior qualidade, aceitabilidade de iterações e imersões sugere-se que sejam criadas as classes dos objetos virtuais correspondentes, as quais podem ser tão sofisticados quanto desejados. Como o SAV é um *framework* a nova classe não alterará o sistema, pois ela será uma classe filha.

O sistema proposto poderá ser adaptado para funcionar com Sistema de Telepresença, pois o sistema de controle, o MCC, é uma ferramenta que pode controlar tanto ambientes virtuais, conforme demonstrado neste trabalho, como poderá efetuar o controle de sistemas reais (Redes de Petri é uma ferramenta consagrada para sistema de controle). Isso seria possível pois teríamos um único módulo central de controle para o AV e para o ambiente Real, por exemplo: Um Robô real pode efetuar os movimentos conforme os comandos efetivados no ambiente virtual, e também se o robô sofrer alguma pane identificada por um sensor que informe o controle, automaticamente refletirá no AV.

O sistema foi implementado com todos os objetos centrados no servidor, mas com suporte para utilizar computação distribuída (objetos distribuídos), assim cada elemento do ambiente pode ser instanciado e processados em máquinas diferentes. Com esta idéia é possível emular sistemas de manufatura sendo que cada elemento (AGV, TORNO, OPERADOR e/ou ROBO) seria uma instância virtual em um computador ou um objeto real, de tal maneira que o MCC não faria a distinção entre os objetos virtuais e objetos reais. Portanto o mundo (virtual ou real) seria único para o modelo.

A previsões para a fábrica do futuro descrita por OLIVEIRA&COELHO (2000) "... imaginar um... a visita ao chão de fábrica, a verificação das condições de trabalho das máquinas, a avaliação do desempenho das ferramentas e a emissão de ordens para correções e melhorias, todos feitos a distância, por Internet. Esse cenário certamente deve ser um símbolo que representa a automação da Fábrica do Futuro, ..." pode ser alcançada com a proposta deste trabalho com um chão de fabrica virtual interligado com um sistema de controle que integra os Avs e o Real.

A construção de um Simulador de Ambiente Virtual (SAV) independente das ferramentas pré-existentes proporcionou maior flexibilidade para adaptar-se a outros sistemas de modelagem, inclusive extensões de RP (RP modular, RP orientada a objetos...) que pode minimizar as limitações do metamodelo.

## 6. Bibliografia

ADAMS (1994) ADAMS, L. (1994). *Visualização e realidade virtual*, Ed. Makron Books, pp. 255-259, São Paulo.

ANGSTER, S. & JAYARAM, S. (1997). *Vedam: virtual environments for design and manufacturing*. [http://www.vrcim.wsu.edu/vir\\_man.html](http://www.vrcim.wsu.edu/vir_man.html) (Agosto).

ARAÚJO, R. B. (1996). *Especificação e análise de um sistema distribuído de realidade virtual*, São Paulo, Junho, 144 p., Tese (Doutorado), Departamento de Engenharia de Computação e Sistemas Digitais, Escola Politécnica da Universidade de São Paulo.

AUTO SIMULATIONS, (1998). *AutoMod User's Manual*, v. 2, Cap. 8, Bountiful, Utah.

BAGIANA, F. (1993). Tomorrow's space: journey to the virtual worlds, *Comput & Graphics*, v.17, n. 6, p. 687-690.

BANERJEE, A. et al. (1997). *Factory Models using virtual reality: Immersive Display Models of factory floor*. <http://zenith2.me.vic.edu/ivri/fmur.htm> (Agosto).

BANKS, J. (1999). Introduction to simulation. *Winter Simulation Conference Proceedings*, p.7-13.

BANKS, J., CARSON, J. S., NELSON B. L.(1995) *Discrete-Event System Simulation*, 2 ed. New Jersey, Prentice Hall.

- BARNERS, M (1996). Virtual Reality and Simulation, In: *Proceeding of the 1996 Winter Simulation Conference*, p.101-110.
- BEL, G.; CAVAILLE, J. B. DUBOIS D. (1988); *Flexible manufacturing systems: modelling and Simulation*. Systems and Control: Encyclopedia, Theory, Technology, Application, p - 1642-1649.
- BISHOP, G. et al. (1992). Research directions in VR environments, *Computer Graphics - ACM*, v. 26, n. 3, p. 153-177, August.
- BLANCHARD, C. et al. (1990). Reality built for two: a virtual reality tool, *Computer Graphics*, v. 24, n. 2, p. 35-36.
- BRUZZONE, G. et al. (1999). Internet mission control of the ROMEÓ Unmanned Underwater Vehicle using the CORAL mission controller. In: *Oceans Conference Record - (IEEE)*, v. 3, p. 1081-1087.
- CECIL, J. A.; et al. (1992). A review of petri-net applications in manufacturing. *The International Journal of Advanced Manufacturing Technology*, v.7, p.168-177. apud SOARES, J. B. (2001). *Editor de modelos de sistemas de eventos discretos, baseado em redes de petri interpretadas*. São Carlos, 2001. 57p. Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.
- CHWIF, L. (1999). *Redução de modelos de simulação de eventos discretos na sua concepção: uma abordagem causal*. São Paulo, 139 p. Tese (Doutorado) – Escola Politécnica, Universidade de São Paulo.
- COMMITTEE ON ADVANCED ENGINEERING ENVIRONMENTS, AERONAUTICS AND SPACE ENGINEERING BOARD, (2000). *Design in the New Millennium: Advanced Engineering Environments: Phase 2*. National Academy of Engineering, National Research Council, National Academy Press, Washington, DC 80 p.
- COMMITTEE ON ADVANCED ENGINEERING ENVIRONMENTS. (1999). *Advanced Engineering Environments: Achieving the Vision, Phase 1*., National Research Council, National Academy Press, Washington, D.C. 58 p.

COMMITTEE ON MODELING AND SIMULATION, (1997). *Modeling and Simulation: Linking Entertainment and Defense*. Computer Science and Telecommunications Board, National Academy Press, Washington, D.C. 1997 196 p.

COMMITTEE ON VISIONARY MANUFACTURING CHALLENGES (1998) *Visionary Manufacturing Challenges for 2020*. Committee on Visionary Manufacturing Challenges, Board on Manufacturing and Engineering Design Commission on Engineering and Technical Systems -National Research Council, National Academy Press, Washington, D.C. 172p.

COMMITTEE TO STUDY INFORMATION TECHNOLOGY AND MANUFACTURING. (1995). *Information Technology for Manufacturing: A Research Agenda*, National Research Council, National Academy Press ,192 p.

CRUZ-NEIRA, C. et al. (1992). The CAVE audio visual experience automatic virtual environment, *Communication of the ACM*, 35(6):64-72, June.

DELMIA (2001) DELMIA. (2001) *Delmia – Make Digital Manufacturing Drive Reality* <http://www.delmia.com/> ( novembro)

DEUTSCH, L. P. (1989) Design reuse and frameworks in the Smalltalk-80 programming system. New York, ACM - Software Reusability, v.2, p. 55-71 apud SOARES, J. B, (2001). *Editor de modelos de sistemas de eventos discretos, baseado em redes de petri interpretadas*. São Carlos, 2001. 57p. Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.

DIVISION (2001). *Solution softwares of virtual reality*. <http://www.ptc.com/products/division/index.htm> (novembro)

DONALD, D. L. et al. (1999) The new design:The changing role of industrial engineers in the design process through the use of simulation. In: *Proceedings of the 1999 Winter Simulation Conference*, p. 829-833

DVORAK, P. (1997). Engineering puts virtual reality to work, *Machine Design*, p. 69-73, February.

- EARNSHAW, R. A. et al. (1995). *Virtual Reality applications*. London, Academic Press, 328 p.
- ECONOMOU, D., MITCHELL, W. L., PETTIFER, S. R., WEST, A. J. (2000). CVE Technology Development Based on Real World Application and User Needs. In: *Proceedings of WET ICE 2000: 9th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, National Institute of Standards & Technology, (NIST), Gaithersburg, Maryland, USA, 14-16 June, IEEE Computer Society Press, p. 12-20.
- ELLIS, S. R. (1994). What are virtual environments?. *IEEE Computer Graphics and Application*, p. 17-22, January.
- ENCARNAÇÃO, J. et al. (1994). European activities in virtual reality. *IEEE Computer Graphics & Applications*, p. 66-74, January.
- ENCARNAÇÃO, L.M. & FRAUNHOFER, M. H. (1998). Technical Report: A Highly Reusable Architecture for Writing Complex Interaction Software. *Virtual Tables Project* Version: 03/02/98, <http://www.crcg.edu/projects/vtforum/contributions/VirtualTables97/index.html>.
- EXHIBITORS (1997). *Virtual reality in manufacturing research and education*. [http://www\\_ivri.me.uic.edu/symp96/preface.html](http://www_ivri.me.uic.edu/symp96/preface.html) (Agosto).
- FISHWICK, P. A. (2000). 3D behavioral model design for simulation and software engineering. In: *Proceedings of the Annual Symposium on the Virtual Reality Modeling Language, VRML*. ACM, New York, NY, USA. p 7-16
- GAMMA, E.; et al. (1997) *Design patterns: elements of reusable object-oriented software*. MA, Addison-Wesley.
- GERMAN, R.; KELLING, C.; ZIMMERMANN, A.; HOMMEL, G.; *TimeNET - a toolkit for evaluating non-Markovian stochastic petri nets*, International Workshop on Petri Nets and Performance Models 1995.IEEE, Los Alamitos, CA, USA. p 210-211.
- GUIMARÃES, J. O. (2001). *Sistemas orientados a objeto*. <http://www.dc.ufscar.br/~jose/courses/soo/soo.zip> (06 de junho.)

- HALE, R.D., Rokonuzzaman, M., Gosine, R.G. (1999). Control of mobile robots in unstructured environments using discrete event modeling. In: Proceedings-of PIE-The-International-Society-for-Optical-Engineering, v. 3838, p. 275-282.
- HYBINETTE, I. E. M. (2000). Interactive parallel simulation environments. PhD apresentada no Georgia Institute of Technology, 2000, 123 pages.
- INAMASU, R. (1995) *Modelo de fms: uma plataforma para simulação e planejamento*. São Carlos, 134 p. Tese (Doutorado) apresentada na EESC – Escola de Engenharia de São Carlos, Universidade de São Paulo.
- INTELLIGENT MANUFACTURING (1995). *Virtual Reality is for real*, v. 1, n. 12. <http://lionhrtpub.com/IM/IM-12-95/IM-12-vr.html> (Dezembro).
- ISHII, H., et al. (1998). Study on design support system for constructing machine-maintenance training environment based on virtual reality technology. In: *Proceedings-of-the-IEEE-International-Conference-on-Systems,-Man-and-Cybernetics-3*. IEEE, Piscataway, NJ, USA, p. 2635-2640.
- JAIN, S. (1999). Simulation in the Next Millennium. In: *Proceeding of the 1999 Winter Simulation Conference*, p.1478-1484.
- JOHNSON, R. (2001) *Frameworks home page*. [http://st-www.cs.uiuc.edu/users/johnson/frameworks.html](http://www.cs.uiuc.edu/users/johnson/frameworks.html) (20 Mar.)
- JOHNSON, R.; FOOTE, B. (1988) Designing reusable classes. *Journal of Object-Oriented Programming*, v. 1, n. 2, p. 22-35. apud SOARES, J. B, (2001). *Editor de modelos de sistemas de eventos discretos, baseado em redes de petri interpretadas*. São Carlos, 2001. 57p. Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.
- JOHNSON, R.; RUSSO, V. (1991) *Reusing object-oriented designs*. United States, Urbana, Department of Computer Science, University of Illinois. (Tech Report UIUCDCS 91-1696) apud SOARES, J. B, (2001). *Editor de modelos de sistemas de eventos discretos, baseado em redes de petri interpretadas*. São Carlos, 2001. 57p. Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.

- JONES, K & CYGNUS, M. W. (1993). Virtual reality for manufacturing simulation. *Proceedings of the Winter Simulation Conference*, New York, IEEE Computer Society Press, p. 882-887.
- KAHANER, D. (1994). Japanese Activities in virtual reality, *IEEE Computer Graphics and Application*, p. 75-78, January.
- KALAWSKY, R. S. (1993). *The science of virtual reality and virtual environments*. Addison-Wesley, 405 p.
- KATO, E. R. R. (1999) . *Ambiente para o projeto de intertravamentos utilizando redes de Petri*. São Carlos, 1999. 172 p. + anexos. Tese (Doutorado) EESC - ESCOLA DE ENGENHARIA DE SAO CARLOS.
- KESAVADAS, T. & SUDHIR, A. (2000). Computational steering in simulation of manufacturing systems, In: *Proceedings-IEEE-International-Conference-on-Robotics-and-Automation*, v.3, IEEE, Piscataway, NJ, USA. p 2654-2658
- KHOSHAFIAN, A. R. (1990) *Object orientation: concepts, languages, databases, user interfaces*. New York, John Wiley & Sons. Apud SOARES, J. B, (2001). *Editor de modelos de sistemas de eventos discretos, baseado em redes de petri interpretadas*. São Carlos, 2001. 57p. Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.
- KIRNER, C. & PINHO (1996) Uma Introdução à Realidade Virtual, *SIBGRAPI - Simpósio Brasileiro de Computação Gráfica e Processamento de Imagens*. Outubro, 1996, Campos do Jordão, SP. Minicurso.
- KIRNER, C. (1996). *Apostila do ciclo de palestras de realidade virtual*, Atividade do Projeto AVVIC- CNPq (Protem - CC - fase III) - DC/UFSCar, São Carlos, p. 1-10, Out. ou <http://www.realidadevirtual.com.br/publicacoes/publicacoes.htm>
- KIRNER, C., (1999). Realidade Virtual: Dispositivos e Aplicações. ESCOLA REGIONAL DE INFORMÁTICA DA SBC REGIONAL SUL, 7. Anais ERI, SBC, Londrina, Chapecó, Novo Hamburgo, maio de 1999, pp. 135-158.

- LEE, K. C.; LU, I. Y.; LIN, H. H.; (1994) PM-Net: a software project management representation model, *Information and Software Technology* v 36 n 5 May 1994. p 295-308.
- LESTON, J. (1996). Virtual reality: the it perspective. *Computer Bulletin*, p. 12-13, June.
- LIN, F., SU, C-J., TSENG, M.M. (1999). An agent-based approach to developing intelligent virtual reality-based training systems. In: *Proceedings 11th International Conference on Tools with Artificial Intelligence*. IEEE Comput. Soc, Los Alamitos, CA, USA, 1999, p.253-60.
- LIU, A.C. and ENGBERTS, A, 1990 "A Petri Net-based Distributed Debugger," *Proceedings of The 14th International Computer Software and Applications Conference*, pp. 639-646, Oct , Chicago, USA.
- LIU, B.; ROBBI, A.; (1995) TiPNet: a graphical tool for timed petri nets, *Inter. Workshop on Petri Nets and Performance Models 1995*. IEEE, Los Alamitos, CA, USA. p 212-213. apud KATO, E. R. R. (1999) . *Ambiente para o projeto de intertravamentos utilizando redes de Petri*. São Carlos, 1999. 172 p. + anexos. Tese (Doutorado) EESC - ESCOLA DE ENGENHARIA DE SAO CARLOS
- LOBÃO, E.C & PORTO, A. J. V. (1996). Evolução das técnicas de simulação em acordo com a tecnologia. In: ENCONTRO NACIONAL DE ENGENHARIA DE PRODUÇÃO – ENEGEP, 16. , Piracicaba. *Anais. Piracicaba*.
- LOFTIN, R. B. & KENNEY, P. J. (1995). Training the hubble space telescope flight team. *IEEE Computer Graphics and Application*, p. 31-37, September.
- MACHADO, L.S. (1995). Conceitos Básicos da Realidade Virtual. *Relatório Técnico*. Instituto Nacional de Pesquisas Espaciais.[INPE- 5975 – PUD/0-25] [www.lsi.usp.br/~liliane/publi.html](http://www.lsi.usp.br/~liliane/publi.html)
- MASCARENHAS, R. et al. (1998) *Modeling and Analysis of a virtual reality system with Time Petri Nets, proceedings of the ISEA Conference, Tokyo, Japan* , p. 20-22, April.

- MCCARTY, W. D. et al. (1994). A virtual cockpit for a distributed interactive simulation, *IEEE Computer Graphics and Application*, p. 49-54, January.
- MEYERS, S. (1992) *Effective C++*. MA, Addison-Wesley.
- MO, J. P.T., TANG, B. C.K. (1998). Petri net modelling and design of task oriented messaging system for robot control. *Computers-and-Industrial-Engineering*, v. 34, n. 4, p. 729-742, Sept.
- MOORE, K. E.; BRENNAN, J. E., (1995). ALPHA/Sim simulation software tutorial. *Winter Simulation Conference Proceedings* . p 387-394.
- MORANDIN Jr., O.(1994); *Projeto e construção de um veículo autoguiado para sistemas flexíveis de manufatura*. Dissertação (Mestrado), EESC – Escola de Engenharia de São Carlos, São Carlos, 123 p.
- MORIE, J. F. (1994). Inspiring the future: merging mass communication, art, entertainment and virtual environment, *Computer Graphics*, 28(2):135-138, May.
- MORTENSEN, K. H. & CHRISTENSEN, S. (2001) *Welcome to the Petri Nets World - Tools and Software*. <http://www.daimi.au.dk/PetriNets/tools> (31 de maio).
- MOSHELL, J. M. et al. (1994). Dynamic terrain. *Simulation*, v. 62, n. 1, p. 29-40, January.
- MURATA, T.(1989) Petri Nets: Properties, Analysis and Applications. *Proceedings of the IEEE*, v. 77, n.4, p. 541– 574.
- NANCE R. E.(1993). A History of Discrete Event Simulation Programming Languages. In: *Proceeding of the Second ACM SIGPLAN History of Programming Languages Conferences*, Cambridge, MA, 28(3), pp. 149-175
- Apud CHWIF, L. (1999). Redução de modelos de simulação de eventos discretos na sua concepção: uma abordagem causal. São Paulo, 139 p. Tese (Doutorado) – Escola Politécnica, Universidade de São Paulo.
- NORMAN, Van B (1992). Future Directions in Manufacturing Simulation. *Industrial Engineering*, p. 36 – 37, Jul.

- O'NEILL, B. (1995). Industry news: Virtual NC for the manufacturing environment. *Simulation Magazine*, p. 330-331, May.
- OLIVEIRA, J. F. G. & COELHO, R. T. (2000) Automação de Chão-de-Fábrica, Virtual in *Fábrica do Futuro: Entenda hoje como sua indústria vai ser amanhã*, Editoras Banas, dez .
- OWEN, J. V. (1994a) Enter the future factory now. *Manufacturing Engineering*, p. 24-25, March.
- OWEN, J. V. (1994b) Making virtual manufacturing real. *Manufacturing Engineering*, p. 33-37, November.
- PENG, Q. (2000). Application and evaluation of VR-based CAPP system. *Journal of Materials Processing Technology*, v. 107, p.153-159.
- Petri Net Tools (2001a) <http://www.petrinet.sc.usp.br> (Maio)
- PIDD, M. (1992) *Computer Simulation in Management Sciences*, John Wiley and Sons, Chischester, 3 th edition.
- PIMENTEL, K. & BLAU, B. (1994). Teaching your system to share, *IEEE Computer Graphics and Application*, pp. 60-65, January.
- POLITANO, P. R.. (1993) *Especificação e implementação de uma unidade de controle para célula flexível de manufatura*. São Carlos, Agosto de 1993, Dissertação de mestrado, Escola de Engenharia de São Carlos - Universidade de São Paulo.
- PORTO, A. J. V.; MORANDIN Jr, O.; POLITANO, P. R.; INAMASU, R. Y.(1993) *Controle de Sistemas de Veículos Auto-Guiados em Sistemas Flexíveis de Manufatura*. Anais do I Encontro Internacional de Instrumentação, Sistemas e Automação Industrial.
- PORTO, A. J. V. & PALMA, J. G. (2000). Manufatura Virtual in *Fábrica do Futuro: Entenda hoje como sua indústria vai ser amanhã*, Editoras Banas, dez .

- PORTE, A. J. V. (1990). *Desenvolvimento de um método de integração do planejamento do processo de fabricação e do planejamento e controle da produção, baseado na flexibilidade do processo de fabricação*, Tese (Doutorado), Escola de Engenharia de São Carlos - USP, São Carlos - SP, 236 Pp..
- PORTE, A. J. V., MIRANDA Jr, J. L. (1998). *Notas de aula de Simulação de Eventos Discretos*: Disciplina oferecida no programa de Pós-Graduação do departamento de Engenharia Mecânica da Escola de Engenharia de São Carlos (EESC), Universidade de São Paulo, São Carlos.
- PRADO, A. F., (2001) Apostila: Sistemas Orientados a Objetos,  
<http://www.dc.ufscar.br/~prado/ensino/>
- PRITSKER (1996) *Introduction to Simulation and Slam*, John Wiley & Sons, 1986.  
apud CHWIF, L. (1999). Redução de modelos de simulação de eventos discretos na sua concepção: uma abordagem causal. São Paulo, 139 p. Tese (Doutorado) – Escola Politécnica, Universidade de São Paulo
- REVISTA DE INFORMAÇÃO E TECNOLOGIA (1997). Aplicações Multimídia, Informativo Técnico no. 23 (21/03/97)  
<http://www.revista.unicamp.br/infotec/informacao/inf23.htm> (17/08/2001).
- RILLO, M.; Controle de Sistema de Manufatura por Redes de Petri e Regras de Produção. *XXII Congresso Nacional de Informática - SUCESSU - anais*- São Paulo, p 677-681. Setembro de 1989.
- ROBERTSON, G. G. et al. (1993). Non immersive virtual reality, *IEEE Computer*, pp. 81-83, Feb.
- ROSEMBLUM, L.J. & MACEDONIA, M.R., (1998). Projects in VR. *IEEE Comput. Graphics Appl.* v.18, n. 6., p. 46-51, novembemr/December.
- SEEVERS, C. (1988). Simulation before automation, In: *Proceedings of the 4th International Conference on Simulation in Manufacturing*, pp. 217-244, November.

- SENSE8 (1996). *Customer applications: information management, analysis, training, simulation, research, education and development*, Sense8 corporation, Sausalito, CA. <http://www.sense8.com/> (Setembro).
- SENSE8 (2001). The Sense8 ProductLine corporation, Sausalito, CA. <http://www.sense8.com/> (novembro).
- SENSE8 Corporation (1997). *WorldToolKit Reference Manual*. Release 7, Mill Valley
- SHIMIT, S. & DUKIE, D. (1999). Virtual environments as hybrid systems. *Proceedings of Eurographics UK 17th Annual Conference: EG-UK99*, Cambridge, UK .
- SHIMIT, S. DUKIE, D. WRIGHT, P. (1999). Using the Resources Model in virtual environment design. In: Workshop on User Centered Design and Implementation of Virtual Environments, University of York, p. 57-72, 30th September.
- SOARES, J. B, 2001. *Editor de modelos de sistemas de eventos discretos, baseado em redes de petri interpretadas*. São Carlos, 2001. 57p. Dissertação (Mestrado) – Escola de Engenharia de São Carlos, Universidade de São Paulo.
- STROUSTRUP, B. (1991). *The C++ Programming Language*. MA, Addison-Wesley.
- SUPERSCAPE INC. (2001). *Interactive 3D solution*. <http://www.superscape.com> (novembro)
- TERESKO, J. (1995). Customers transform virtual prototyping. *IW Eletronics & Technology*, p. 35-37, May.
- VALÉRIO NETTO, A. (1998) Prototipação de um torno CNC utilizando realidade virtual, São Carlos, 1998. 115 p. + anexos. Dissertação (Mestrado) EESC - ESCOLA DE ENGENHARIA DE SAO CARLOS.
- VALÉRIO NETTO, A., PORTO A. J. V. , PALMA, J. G. (1997). A utilização da realidade virtual na engenharia mecânica, *CONGRESSO BRASILEIRO DE ENGENHARIA MECÂNICA - COBEM , 14. Anais*, Bauru, Dez.

- VON SCHWEBER, L. & VON SCHWEBER, E. (1995). Cover story: realidade virtual. *PC Magazine Brasil*, v. 5, n. 6, p. 50-73, junho.
- WANG, L.C. (1996). Object-Oriented Petri nets for modelling and analysis of automated manufacturing systems. *Computer Integrated Manufacturing Systems*, v. 26, n.2, p.111-125.
- WILLANS, J. S., MICHAEL D. H., SMITH, S. P. (2001). Implementing virtual environment object behaviour from a specification, *User Guidance in Virtual Environments: Proceedings of the Workshop on Guiding Users through Interactive Experiences - Usability Centred Design and Evaluation of Virtual 3D Environments*, Shaker Verlag, Aachen, Germany, p. 87-97
- YOSHIKAWA, H., TEZUKA, T., KASHIWA, K. I., ISHII, H. (1997). Simulation of machine-maintenance training in virtual environment. *Nippon-Genshiryoku-Gakkaishi-Journal-of-the-Atomic-Energy-Society-of-Japan*. V. 39, n. 12, p 1078-1089, Dec./ Resumo em Compendex.
- ZHANG, Y., SEHGAL, A., VOLZ, R. (1999). Petri Net Driven Graphical Simulation Tool. ANRPCP technical.
- ZHANG, Y., VOLZ, R., SEHGAL, A. (2000). Generating Perfi Net Driven Graphical Simulation Tool for Automated Systems. In: *Proceedings of the American Nuclear Association 2000 Annual Meeting*, p.140-147.
- ZHOU, Y., MURATA, T., DEFANTI, T., ZHANG, H. (2000a) Modeling and performance analysis using extended fuzzy-timing Petri nets for networked virtual environments. *IEICE-Transactions-on-Fundamentals-of-Electronics,-Communications-and-Computer-Sciences*, v E83-A, n. 11, p 2166-2176, Nov.
- ZHOU, Y., MURATA, T., DEFANTI, T., ZHANG, H. (2000b). Fuzzy-timing Petri net modeling and simulation of a networked virtual environment. *IEICE-Transactions-on-Fundamentals-of-Electronics,-Communications-and-Computer-Sciences*, v. E83-A, n. 11, p 2166-2176, Nov.

## Apêndice A - Propriedades e métodos de análise da Rede de Petri

Propriedades e métodos de análise da Rede de Petri levantadas por MURATA (1989) em um artigo especial (*Invited paper*) para o *Proceedings* da IEEE (revisão de RP com mais de 300 referências) e traduzida por SOARES (2001).

### **6.1.1.1 Propriedades comportamentais**

As propriedades comportamentais da rede de Petri indicam como o sistema se comporta a partir da marcação inicial. Segue abaixo algumas propriedades comportamentais:

- **Alcançabilidade**

A alcançabilidade é base fundamental para o estudo de propriedades dinâmicas de qualquer sistema, e consiste em determinar se um dado estado  $M_n$  é alcançável a partir de uma seqüência de disparos de transições, numa rede  $(N, M_0)$ . Isto é, determinar se  $M_n \in R(M_0)$ . Esta seqüência de disparos é denotada por  $\sigma = M_0 t_1 M_1 t_2 M_2 \cdots t_n M_n$  ou simplesmente por  $\sigma = t_1 t_2 \cdots t_n$ .

- **Limitabilidade**

Uma rede de Petri  $(N, M_0)$  é dita ser k-limitada ou simplesmente limitada se o número de marcas em cada lugar não exceder um número finito k para qualquer marcação alcançável de  $M_0$ , isto é,  $M(p) \leq k$ , para cada lugar  $p$  e para cada marcação  $M \in R(M_0)$ . Uma rede de Petri é dita ser segura se é 1-limitada.

Os lugares em rede de Petri são muitas vezes usados para representar buffers ou registros para armazenamento de dados de intermédio. Verificar se uma rede é limitada ou segura, garante que não haverá estouro nos buffers ou registros (overflows), independentemente da seqüência sendo disparada.

- **Vivacidade**

Uma rede de Petri  $(N, M_0)$  é tida como viva se para qualquer marcação  $M \in R(M_0)$ , existir sempre pelo menos uma transição habilitada para disparo. A vivacidade é uma propriedade ideal para muitos sistemas. No entanto, em muitos casos ela é impraticável sendo muito caro a verificação dessa propriedade, como por exemplo na modelagem de sistemas operacionais de sistemas computacionais robustos. Neste sentido são definidos diferentes níveis de vivacidade. Assim, uma transição  $t$  numa rede de Petri  $(N, M_0)$  é dita ser:

morta ( $L_0$ -viva) se  $t$  nunca pode ser disparada em qualquer seqüência de disparo em  $L(M_0)$ ;

$L_1$ -viva (potencialmente disparável) se  $t$  pode ser disparada pelo menos uma vez em alguma seqüência de disparo em  $L(M_0)$ ;

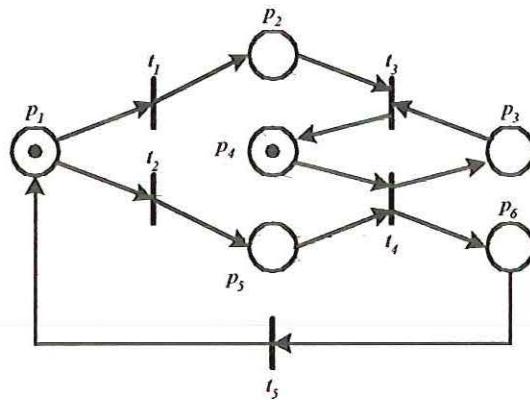
$L_2$ -viva se, dado um inteiro positivo  $k$ ,  $t$  pode ser disparado pelo menos  $k$  vezes em alguma seqüência de disparo em  $L(M_0)$ ;

$L_3$ -viva se  $t$  aparece muitas vezes em alguma seqüência de disparo em  $L(M_0)$ ;

$L_4$ -viva se  $t$  é  $L_1$ -viva para cada marcação  $M \in R(M_0)$ .

Uma rede de Petri  $(N, M_0)$  é dita com  $L_k$ -viva se cada transição na rede for  $L_k$ -viva, para  $k = 0, 1, 2, 3, 4$ .

A Figura 5 apresenta uma rede de Petri não viva, uma vez que nenhuma transição fica habilitada para disparo se a transição  $t_1$  for disparada inicialmente. No entanto ela pode ser classificada como estritamente  $L_1$ -viva.



**Figura 1.** Um exemplo de rede de Petri segura, não viva, mas estritamente L1-viva.

- **Reversibilidade e estado origem**

Uma rede de Petri  $(N, M_0)$  é dita reversível se, para cada  $M \in R(M_0)$ ,  $M_0$  é alcançável de  $M$ . Desta forma, numa rede reversível é sempre possível retornar ao estado inicial. Em muitas aplicações não é necessário retornar ao estado inicial, sendo suficiente retornar a algum estado (origem). Uma marcação  $M'$  é dita estado origem se, para cada  $M \in R(M_0)$ ,  $M'$  é alcançável de  $M$ .

- **Abrangência de cobertura**

A marcação  $M$  numa rede de Petri  $(N, M_0)$  é dita ser abrangível de cobertura se existir uma marcação  $M' \in R(M_0)$  tal que  $M'(p) \geq M(p)$  para cada  $p$  da rede. Se uma marcação  $M$  for a marcação mínima para o disparo de uma transição  $t$ , então  $t$  é morta (não L1-viva) se e somente se  $M$  não for abrangível de cobertura.

- **Persistência**

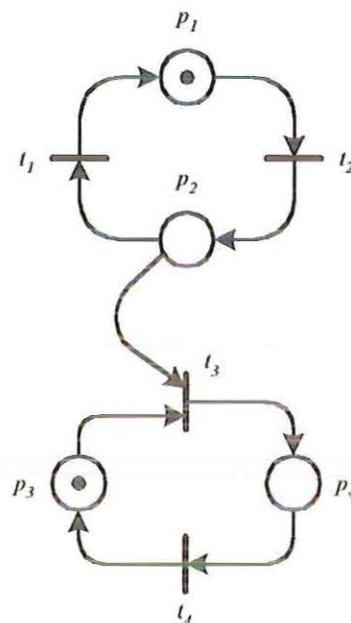
Uma rede de Petri  $(N, M_0)$  é dita persistente se, para qualquer duas transições habilitadas, o disparo de uma não desabilita a outra. Uma transição numa rede persistente, uma vez habilitada, continuará habilitada até seu disparo. A noção de persistência é muito útil no contexto de sistemas paralelos.

- **Distância sincrônica**

O conceito de distância sincrônica está relacionado a um grau de dependência mútua entre dois eventos em um sistema condição/evento. A distância sincrônica entre duas transições  $t_i$  e  $t_j$  numa Rede de Petri ( $N, M_0$ ) é definida por:

$$d_{ij} = \max |\bar{\sigma}(t_i) - \bar{\sigma}(t_j)|,$$

onde  $\sigma$  é uma seqüência de disparo a partir de alguma marcação  $M \in R(M_0)$  e  $\bar{\sigma}(t_i)$  é o número de vezes que a transição  $t_i$  dispara em  $\sigma$ . Por exemplo, a Figura 6 mostra uma rede no qual  $d_{12} = 1$ ,  $d_{34} = 1$  e  $d_{13} = \infty$ .



**Figura 2.** Exemplo para verificação da distância sincrônica.

#### 6.1.1.2 Métodos de análise

A partir de uma análise do modelo pode-se determinar todos os estados que o sistema pode assumir, e como estes estados alternam-se entre si de acordo com eventos relacionados ao sistema. Desta forma, consistem em ferramentas fundamentais na validação do modelo.

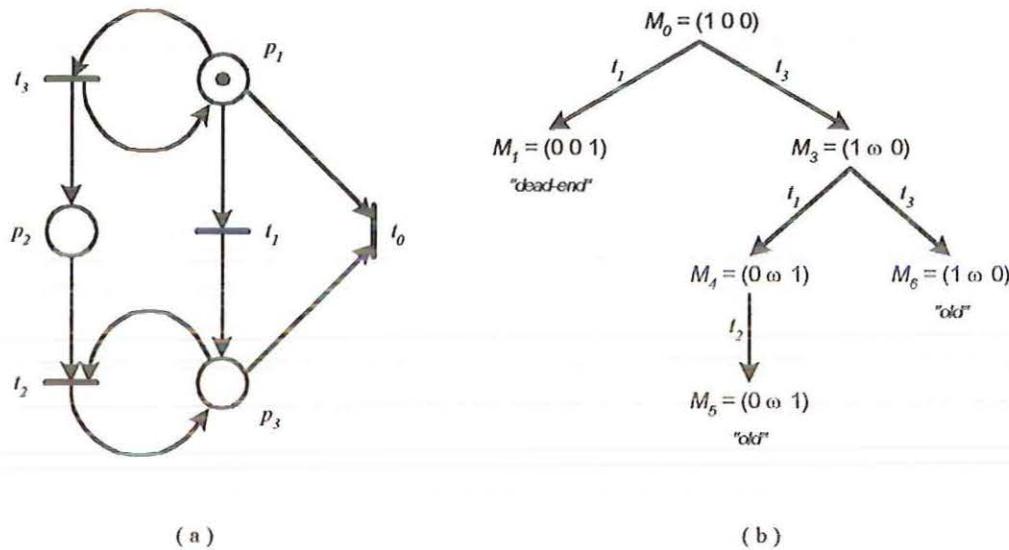
- **Árvore de alcançabilidade**

A árvore de alcançabilidade descreve todos os possíveis estados do sistema e a seqüência de disparos para alcançá-los. Dada uma Rede de Petri ( $N, M_0$ ), a partir

da marcação inicial  $M_0$  pode-se obter outras marcações de acordo com o disparo das transições habilitadas. Esse processo pode ser interpretado como uma árvore de marcações. Cada nó representa uma marcação a partir de  $M_0$  (nó raiz), e cada arco (galho) representa o disparo de uma transição.

Em redes não limitadas, essa representação em árvore cresceria infinitamente. Assim, introduz-se o símbolo  $\omega$ , representando uma marcação que tende ao infinito. Dessa forma, para cada inteiro  $n$ ,  $\omega > n$ ,  $\omega \pm n = n$  e  $\omega \geq \omega$ .

A Figura 7 exibe um exemplo de representação da árvore de alcançabilidade para uma rede de Petri. Nota-se que é introduzido o conceito de *dead-end*, indicando uma marcação no qual não há transições habilitadas, e *old*, indicando que a marcação já existe em algum nó de nível inferior da árvore (mais próximo da raiz).



**Figura 3.** Exemplo de representação da árvore de alcançabilidade

- **Matriz de incidência e equação de estado**

A *matriz de incidência* e a *equação de estado* possibilitam a análise algébrica do comportamento dinâmico das redes de Petri.

Para uma Rede de Petri  $(N, M_0)$  composta por  $n$  transições e  $m$  lugares, a matriz de incidência  $A = [a_{ij}]$  é uma matriz  $n \times m$  de inteiros onde,

$$a_{ij} = a_{ij}^+ - a_{ij}^- ,$$

$a_{ij}^+ = W(i, j)$  é o peso do arco da transição  $i$  para o lugar  $j$  e

$a_{ij}^- = W(j, i)$  é o peso do arco do lugar  $j$  para a transição  $i$ .

Observa-se que  $a_{ij}^+$ ,  $a_{ij}^-$  e  $a_{ij}$  representam, respectivamente, o número de marcas adicionadas, removidas e alteradas no lugar  $j$  a partir do disparo da transição  $i$ . Uma transição  $i$  está habilitada para disparo numa marcação  $M$  se e somente se:

$$a_{ij}^- \leq M(j), \quad j = 1, 2, \dots, m.$$

A equação de estado indica a alteração de estado (marcação) numa rede de Petri, após o disparo de uma transição. Para uma Rede de Petri  $(N, M_0)$  composta por  $n$  transições,  $m$  lugares e uma matriz de incidência  $A$ , define-se a equação de estado como:

$$M' = M + A^T u \text{ onde,}$$

$M$  e  $M'$  representam, respectivamente, a marcação antes e depois do disparo;

$A$  é a matriz de incidência;

$u$  é um vetor coluna  $n \times 1$  de controle, de  $n-1$  posições de valor zero e uma entrada de valor um, indicando a transição disparada.

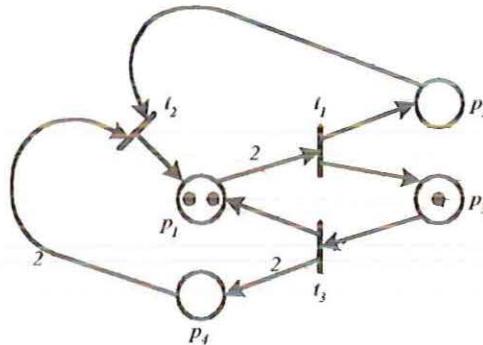


Figura 4. Um exemplo de rede de Petri.

Por exemplo, considerando a rede de Petri da Figura 8, com marcação inicial  $M_0 = (2 \ 0 \ 1 \ 0)^T$ , a equação de estado pode ser utilizada para determinar qual a marcação da rede, a partir da e do disparo da transição  $t_3$ :

$$M' = \begin{bmatrix} 2 \\ 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -2 & 1 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & -2 & 2 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 3 \\ 0 \\ 0 \\ 2 \end{bmatrix}$$

Considerando a equação de estado e uma seqüência de  $d$  disparos a partir da marcação inicial  $M_0$ , pode-se escrever a seguinte equação:

$$M_f = M_0 + A^T \sum_{k=1}^d u_k$$

- **Análise usando invariantes**

A análise de rede de Petri usando invariantes consiste em determinar os *S-invariantes* e *T-invariantes*. Considerando uma Rede de Petri  $(N, M_0)$  composta por  $n$  transições,  $m$  lugares e uma matriz de incidência  $A$ , definem-se:

Um *S-invariante* consiste num vetor  $y$  ( $m \times 1$ ), de inteiros positivos, de forma que  $Ay = 0$ , e assim,  $M^T y = M_0^T y$ , para qualquer  $M \in R(M_0)$ . Esta equação implica que o total de marcas iniciais em  $M_0$ , pesado pelo *S-invariante*, é constante para qualquer seqüência de disparo.

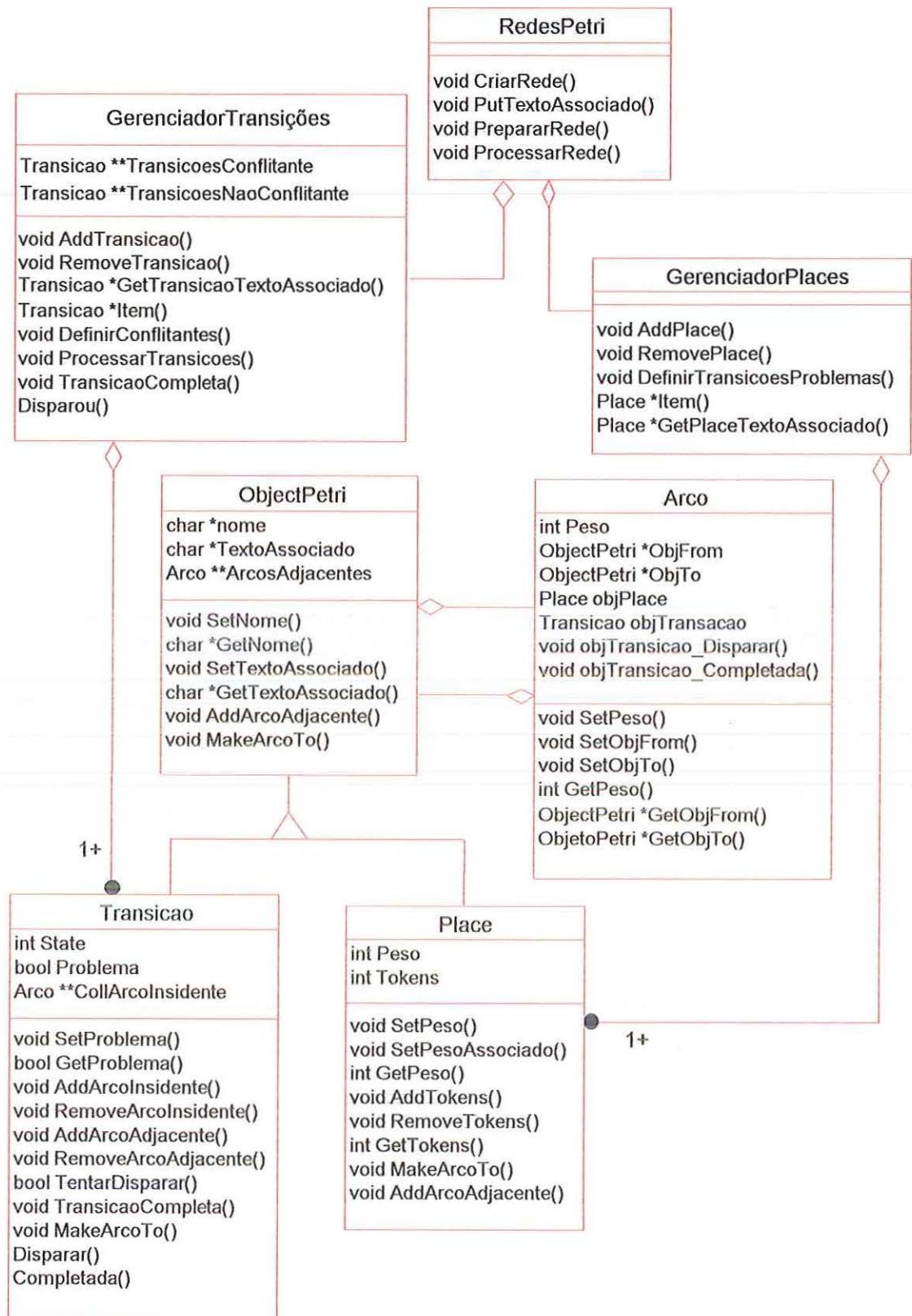
Um *T-invariante* consiste num vetor  $x$  ( $n \times 1$ ), de inteiros positivos, de forma que  $A^T x = 0$ , e assim,  $M_0 = M_f$  para  $x$  sendo a seqüência de disparos. Esta equação implica que se o vetor de disparo for um *T-invariante*, então o sistema retorna para o estado inicial após os disparos.

## **Apêndice B - Ferramentas de Redes de Petri**

A Tabela de ferramenta de Redes de Petri elaborado por MORTENSEN & CHRISTENSEN (2001).

em 20.9.2001	Características												Tipo de RP						Plataforma					Distribuição						
	Graph. editor			Token game			Perform. analysis			Struct. analysis			Transformation			Time	Stochastic	Color	Queue	Priority	Probability	Hierarchy	Win	DOS	Unix	Mac	Other	Free ware	Free educ.	Free trial
	X	X	X	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-			
ALPHA/Sim	X	X	X	-	-	-	X	-	-	-	-	-	-	-	-	X	-	X	-	-	-	-	-	-	-	-				
ARTIFEX	X	X	X	-	-	-	X	-	-	-	-	-	-	-	-	X	-	X	-	-	-	-	-	-	X	-				
CABERNET	X	X	-	X	X	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	X	-	-	X	-				
CodeSign	X	X	-	-	-	-	X	-	-	-	-	-	-	-	-	X	-	X	X	-	-	-	X	-	-					
DaNAMiCS	X	X	X	X	-	-	X	-	X	-	-	-	-	-	-	X	X	-	X	-	-	-	-	-	X	-				
DESIGN/CPN	X	X	X	X	-	-	X	-	X	-	-	-	-	-	-	X	-	X	X	-	-	X	-	-	X	-				
DESIGN/OA	-	X	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	X	X	X	-	-	-	-	-	-				
DNA.net	X	X	X	X	-	-	X	X	-	-	-	-	-	-	-	X	-	X	-	-	X	-	-	X	-					
DSPNExpress	X	X	X	X	-	-	X	X	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-					
EDS Petri Net Tool	X	X	X	X	-	-	X	X	-	-	-	-	-	-	-	X	-	-	-	OS/2	-	-	-	-	-					
ELSIR	X	X	X	X	-	-	X	X	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-					
ExSpect	X	X	X	X	-	-	X	X	-	-	-	-	-	-	-	X	-	X	-	-	-	-	-	-	X	-				
F-net	X	X	X	X	-	-	X	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	-					
GreatSPN	X	X	X	X	X	-	X	X	X	-	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-					
HiQPN-Tool	X	X	X	X	-	-	X	X	X	X	-	-	-	-	-	-	-	-	X	-	-	X	-	-	X	-				
INCOME	X	X	-	-	-	-	-	-	X	-	-	-	-	-	-	X	X	X	-	-	-	-	-	-	X	-				
Leu Smart	X	X	X	X	-	-	X	X	X	X	X	X	X	X	X	X	-	-	-	-	-	-	-	-	X	-				
Looping	X	X	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	X	-				
MACROTEC	X	X	X	X	-	-	-	-	-	-	-	-	-	-	-	X	-	-	X	-	-	-	-	-	-	-				
MISS-RdP	X	X	-	-	-	-	X	X	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-				
MOBY	X	X	X	X	-	-	X	-	X	-	-	-	-	-	-	X	-	X	X	X	OS/2	-	-	X	-					
Moses Tool Suite	X	X	-	-	-	-	X	X	-	-	-	-	-	-	-	X	-	-	-	-	Java	-	-	X	-					
Netmate	X	X	-	X	-	-	X	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-	-	-	X	-				
PACE	X	X	-	-	X	-	X	X	-	-	-	-	-	-	-	X	-	X	X	-	-	-	-	-	X	-				
PAPETRI	X	X	-	X	X	-	X	-	X	-	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-					
Patrice's PN Java Applet	X	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	Java	-	-	X	-					
PDS	X	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-					
PENECA PNC	X	X	-	-	-	-	X	X	-	-	-	-	-	-	-	X	-	X	-	-	-	-	-	-	X	-				
PEP	X	X	-	X	X	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-					
PESIM	X	X	X	X	X	-	X	X	-	-	-	-	-	-	-	X	-	-	-	-	X	-	-	X	-					
Petri Maker	X	X	X	X	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-	-	X	-					
Petri Net Browser	X	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	Java	-	-	X	-					
PetriTool	X	X	-	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	Java	-	-	X	-					
Pn nice	X	X	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-					
PNS	X	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	Java	-	-	X	-					
PNSim	X	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	Java	-	-	X	-					
PNTalk	X	X	X	-	-	-	X	-	-	-	-	-	-	-	-	X	-	X	-	-	-	-	-	X	-					
Product Net Machine	X	X	-	X	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	Lisp	-	-	X	-					
PSITool	X	X	-	X	-	-	X	X	X	-	-	-	-	-	-	-	-	-	X	-	-	-	-	-	-					
QPN-Tool	X	X	X	X	-	-	X	X	X	X	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-					
Renew	X	X	-	-	-	-	X	-	-	-	-	-	-	-	-	X	-	-	-	-	Java	-	-	X	-					
SEA	X	X	-	-	-	-	X	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-					
SYSTEMSPECs	X	X	X	X	-	-	-	-	-	-	-	-	-	-	-	X	-	X	X	X	-	-	-	-	-					
TemPRO	X	X	X	-	-	-	X	-	X	-	X	-	X	-	X	-	X	-	-	-	-	-	-	-	-					
TimeNET	X	X	X	X	-	-	X	X	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-					
VIPtool	X	X	-	-	-	-	-	-	-	-	-	-	-	-	-	X	-	X	-	-	-	-	-	X	-					
Visual Object Net++	X	X	X	-	-	-	X	-	-	-	X	-	X	-	X	-	X	-	-	-	-	-	-	-	X	-				
Visual SimNet	X	X	X	X	-	-	X	X	X	X	X	X	X	X	X	-	-	X	-	-	-	-	X	-	-					
WebSPN	X	X	X	-	-	-	X	X	-	-	-	-	-	-	-	-	-	-	-	Java	-	-	X	-						
WINPETRI	X	X	-	X	-	-	-	-	-	-	-	-	-	-	-	X	-	-	-	-	X	-	-	X	-					

## Apêndice C - Classes do sistema de controle



# Glossário

## Ambiente Virtual (AV)

Ambiente sintético desenvolvido em computador para simular tanto um ambiente imaginário quanto um ambiente real.

## Ambientes Interativos

Ação que se exerce mutuamente entre dois ou mais ambientes

## Ambiente Iterativo

Ambiente que detecta as entradas do usuário e modifica instantaneamente o mundo virtual e as ações sobre ele (capacidade reativa).

## CAVE

O ambiente CAVE (Cave Automatic Virtual Environment), é baseado em um cômodo, onde as paredes, piso e teto são telas que recebem a projeção sincronizada das partes de um mundo virtual.

## Cliente/Servidor

O modelo cliente/servidor identifica a relação entre dois tipos de aplicações, as aplicações clientes e as aplicações servidoras. A aplicação cliente é aquela que requer serviços a outras aplicações, ao invés, a aplicação servidora é aquela que disponibiliza um conjunto de serviços a aplicações clientes. O browser é um exemplo de uma aplicação cliente que requer serviços a uma aplicação servidora de Web server.

## Código

é um conjunto de regras que permite a representação da informação. Este termo aplicado à informática tem diversos significados: pode representar um conjunto de símbolos, possivelmente com significados secretos; pode indicar a colocação de informação em forma de código; pode ainda, significar instruções de programas de computador. Esta palavra aparece associada a diversos conceitos, sendo de destacar: Código de acesso, código ASCII, código assembly, código de barras, código binário, código de país, código máquina, código objecto, código fonte e código de cor.

## Compatibilidade

Capacidade de dois dispositivos de hardware (por exemplo, computador e impressora) ou dois tipos de software funcionarem em conjunto.

## Endereço

Identificação da localização de um dispositivo ou área de armazenamento. Por exemplo: registo de memória ou setores de disco. Em informática, o termo endereço é utilizado em diversas situações, sendo de destacar as seguintes: endereço barramento, endereço de e-mail, endereços de I/O, endereço de Internet, endereço IP.

## **Endereço IP**

É um conjunto de algarismos, agrupados em classes, que permite identificar quem solicita e quem envia dados via Internet, por exemplo, 120.12.3.50. Qualquer site tem um endereço IP próprio. Geralmente atribui-se um domínio ao endereço IP de um site, por forma a facilitar a utilização da Internet. No que respeita aos acessos à Internet, um Endereço IP é atribuído aquando do estabelecimento da ligação, podendo esta atribuição ser feita aleatoriamente pelo ISP, neste caso estamos perante uma ligação com IP Dinâmico, ou através da utilização de um Endereço IP Fixo, em que é atribuído sempre o mesmo endereço para todas as ligações efetuadas.

## **Feedback**

Mudança como resposta à manipulação de um dispositivo de entrada de dados. Dispositivos de saída que oferecem feedback transmitem pressão, força ou vibração para fornecer ao participante da RV a sensação do tato. O feedback tátil simula a sensação ligada à pele. O feedback de força simula peso ou resistência ao movimento.

## **FTP**

**File Transfer Protocol**

Protocolo de Transferência de Arquivos (dados). Protocolo cliente/servidor que permite a transferência de dados de um computador para outro, através de uma rede TCP/IP.

## **Grafical User Interface - GUI**

Trata-se de um interface gráfico direcionado para o usuário, ou seja, um programa de interface que tira partido das capacidades gráficas do computador para tornar o programa mais fácil de utilizar. Um GUI bem concebido liberta o usuário da necessidade de aprender comandos complexos.

## **HMD (head-mounted display)**

Óculos ou capacetes com pequenos monitores que emitem imagens, posicionando cada monitor diante de cada olho do usuário.

## **Hosting**

Termo vulgarmente empregado para definir um Serviço de Alojamento de Aplicações, Páginas, Scripts, entre outros.

## **Imersão** - A idéia de imersão está ligada com o sentimento de se estar dentro do ambiente.

Normalmente, um sistema imersivo é obtido com o uso de capacete de visualização, mas existem também sistemas imersivos baseados em salas com projeções das visões nas paredes, teto, e piso. Além do fator visual, os dispositivos ligados com os outros sentidos também são importantes para o sentimento de imersão, como som, posicionamento automático da pessoa e dos movimentos da cabeça, controles reativos, etc. A visualização tridimensional através de monitor é considerada não imersiva.

## **Interação**

Ação que se exerce mutuamente entre duas ou mais coisas, duas ou mais pessoas (Dicionário Aurélio Básico da Língua Portuguesa, 1988); Influência mútua ou intercâmbio.

## **Internet**

A melhor demonstração real do que é uma auto-estrada da informação. A Internet é uma imensa rede de redes que se estende por todo o planeta e praticamente todos os países. Os meios de ligação dos computadores desta rede são variados, indo desde rádio, linhas telefônicas, ISDN, linhas digitais, satélite, fibras-ópticas, etc. Criada em 1969 pelo Departamento de Defesa dos EUA (DoD) como um projeto pioneiro de constituição de uma rede capaz de sobreviver a ataques nucleares, foi-se expandindo até chegar ao tamanho e importância que hoje.

## **Inteligência Artificial**

É o estudo de aparelhos e sistemas computacionais feitos pelo homem que podem agir de uma maneira que nos levaria a qualificar como inteligente. Esta definição incorpora o aprendizado e a adaptabilidade como características gerais da inteligência.

## **Intuitivo**

É muito importante compreender o conceito de iteração intuitiva. Quando um utilizador toma decisões sobre como navegar no ambiente, essas decisões são influenciadas por informações do mundo real. O termo "intuitivo" significa que se pressupõe que a maioria dos utilizadores tome as mesmas decisões quando confrontada com um determinado elemento do ambiente.

## **IP**

### **Internet Protocol**

Protocolo que permite o envio de dados de um computador para outro através da Internet. Qualquer tipo de operação realizada via Internet é efetuada com base no Endereço IP dos intervenientes, como por exemplo, o envio de uma mensagem de correio eletrônico ou o acesso a um site.

## **Iteração**

*Alg.* Processo de resolução de uma equação mediante uma seqüência de operações em que o objeto de cada uma é o resultado da que a precede (Dicionário Aurélio Básico da Língua Portuguesa, 1988)

Computacionalmente em RV, a idéia de iteração é a mesma, processo de resolução de um Ambiente Virtual mediante uma seqüência de ações dos usuários em que o Ambiente em um dado momento é o resultado da ação que a precede. Está ligada com a capacidade do computador detectar as entradas do usuário e modificar instantaneamente o mundo virtual e as ações sobre ele (capacidade reativa).

## **Iterativo**

Que serve para iterar. Iterável é um adjetivo que significa que pode ou deve ser iterado.

## **Memória**

Local onde os dados e os programas são armazenados num computador. Os dados podem ser armazenados em chips de memória semi-condutores ou em dispositivos de armazenamento de massa. A RAM (Random Access Memory) - memória de acesso aleatório, guarda informação que pode ser gravada e lida por diversas vezes. Estes dados perdem-se quando a alimentação do computador é cortada. A ROM (Read Only Memory) - memória só de leitura, guarda informação que é acedida em modo de leitura e os dados não podem ser apagados.

## **Multimídia**

Combinação de dois ou mais meios (som, vídeo, animação e gráficos) numa aplicação. As grandes quantidades de informação que os arquivos multimídia utilizam, obriga à utilização de dispositivos de armazenamento de alta capacidade.

### **Multitarefa**

Correr mais do que uma aplicação ao mesmo tempo. Capacidade de um computador mudar entre aplicações sem as fechar ou copiar informação entre elas.

### **Mundo virtual Mundo digital.**

"Mundo" criado a partir de técnicas de computação gráfica. Através de dispositivos de saída de dados que estimulam os sentidos do participante, este mundo transforma-se num ambiente de RV.

### **On-line**

Atividade/Transacção realizada em tempo real através de uma ligação à Internet.

### **Orientação a Objetos (OO)**

É um meio para se descrever os sistemas do mundo real. Um objeto representa uma abstração de uma entidade do mundo real, combinando, em um mesmo elemento, informação e comportamento. Os paradigmas anteriores do desenvolvimento de software ligavam apenas fracamente a informação e o comportamento. A orientação a objetos conduz a uma união mais bem definida destes dois conceitos.

### **Path**

Localização exata de um arquivo no computador. Path é utilizado para indicar o local onde o sistema operacional pode encontrar os arquivos, uma vez que a localização se encontra armazenada em variáveis de ambiente do sistema.

### **Pixel**

Pixel é a mais pequena unidade lógica de informação visual utilizada para a construção de uma imagem. Pequenos quadrados que quando uma imagem é aumentada conseguem ser visualizados.

### **Programação**

Concepção e escrita de um programa de computador. O programador analisa o que o programa tem de fazer e escreve o código necessário utilizando linguagens de programação que posteriormente são traduzidas pelo computador para linguagem máquina de forma a interpretar e executar o programa do seu microprocessador.

### **Protocolo**

Conjunto de regras que todos os computadores têm que cumprir para poderem desenvolver tarefas compatíveis entre si. Entre os protocolos mais importantes, contam-se os protocolos de comunicações, sem os quais os computadores não conseguem transferir informações de uns para os outros.

### **Prototipação Virtual**

Geração de protótipos no computador para apresentações e emulações realistas, que permite interações com o produto até mesmo nos estágios iniciais de desenvolvimento.

### **Realidade Virtual (RV)**

Simulação do mundo real (visão, som, sensações tácteis...) por processos inteiramente controlados através de meios eletrônicos.

## **Rede**

Dois ou mais computadores ligados por meio de cabos ou modem (através da linha telefônica) com o intuito de partilharem informação, assim como partilharem dispositivos periféricos (impressoras, scanners, etc.). Um sistema de rede é classificado de acordo com a sua extensão geográfica: LAN (Local Area Network), MAN (Metropolitan Area Network), WAN (Wide Area Network), de acordo com o tipo de protocolo utilizado. É frequente a utilização da expressão "A Rede" (The Net) como referência à Internet.

## **Rendering**

Processo de criação de imagens a partir de modelo. Utiliza técnicas de computação gráfica para o cálculo e desenho dessas imagens.

## **Resolução**

Qualidade de imagem num monitor ou numa impressora. O número máximo de pixels que podem ser exibidos no monitor é expresso como número de pixels horizontais \* números de pixels verticais, por exemplo, 800\*600. A resolução nas impressoras é medida em dpi (dots per inch).

## **Script**

Sequência estruturada de instruções de programação que se encontram num computador servidor e que são interpretadas por uma aplicação específica, que pode ou não residir no servidor. Por exemplo, um script de Javascript é interpretado pelo browser, ao invés, um script de Perl é interpretado pelo interpretador de Perl residente no servidor, e só então o resultado da interpretação será conhecido pelo utilizador, através do browser. As linguagens de script não são compiladas, mas sim interpretadas ao nível do computador servidor sempre que a página que a contenha seja solicitada, o que a torna mais lenta relativamente a programas que já se encontram compilados.

## **Servidor ou Server**

Computador ligado em rede que fornece serviços a outros computadores como: o armazenamento de dados, transferência de arquivos, correio electrónico e World Wide Web.

## **Simulação**

o processo pelo qual os resultados são emitidos em funções de perguntas "E SE", não otimizando o processo efetivamente, mas fornecendo subsídios para as tomadas de decisão dos gerentes.

## **Simulação de eventos discretos**

são modelos discretos aqueles em que o avanço da contagem de tempo na simulação se dá na forma de incrementos cujos valores podem ser definidos em função da ocorrência dos eventos ou pela determinação de um valor fixo, nesses casos só é possível determinar os valores das variáveis de estado do sistema nos instantes de atualização da contagem de tempo;

## **Simulação de processos contínuos**

o avanço da contagem de tempo na simulação dá-se de forma contínua, o que possibilita determinar os valores das variáveis de estado a qualquer instante.

## **TCP/IP**

Transmission Control Protocol/Internet Protocol

É um pacote de protocolos de comunicação utilizado para a conexão de máquinas hospedeiras à Internet. O TCP/IP utiliza vários protocolos sendo os principais o TCP e o IP. O TCP/IP está altamente enraizado no sistema operacional UNIX o que o tornou no padrão de fato para a transmissão de dados em redes.

### **Tempo de latência**

Intervalo de tempo entre um movimento executado pelo usuário e o resultado deste movimento.

### **Tempo real**

Pouco ou nenhum atraso no tempo de resposta, dando a impressão de resposta instantânea.

### **Universo**

Pode parecer muito grandioso, mas o universo é o número de potenciais objetos virtuais num determinado meio ou ambiente, o qual é um importante conceito para avaliar os objetos se comportam como um todo.

**Virtual** - vem do latim medieval *virtualis*, derivado de *virtus*, força, potência. Refere-se ao que não existe como realidade, mas sim como potência ou faculdade, porém sem exercício ou efeito atual .Filosoficamente diz-se do que está predeterminado e contém todas as condições essenciais para a sua realização. Qualquer coisa aparente sem definição prefixada que se contrapõe com o real ou o real absoluto. No uso corrente, a palavra virtual é empregada com freqüência para significar a pura e simples ausência de existência, a "realidade" supondo uma efetuação material, uma presença tangível.

Virtual é como o complexo problemático, o nó de tendências ou de forças que acompanha uma situação, um acontecimento, um objeto ou uma entidade qualquer, e que se chama um processo de resolução: a atualização. Esse complexo problemático pertence à entidade considerada e constitui inclusive uma de suas dimensões maiores.

LÉVY, Pierre. O que é virtual ? São Paulo : Editora 34,1996.