UNIVERSIDADE FEDERAL DE SANTA CATARINA

CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

DESENVOLVIMENTO DE TEMPLATES PARA MODELAGEM, SIMULAÇÃO E AVALIAÇÃO DE DESEMPENHO EM COMPUTADORES COM ARQUITETURA PARALELA

Adhemar Maria do Valle Filho

FLORIANÓPOLIS

1997

UNIVERSIDADE FEDERAL DE SANTA CATARINA

CURSO DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

DESENVOLVIMENTO DE TEMPLATES PARA MODELAGEM, SIMULAÇÃO E AVALIAÇÃO DE DESEMPENHO EM COMPUTADORES COM ARQUITETURA PARALELA

por

Adhemar Maria do Valle Filho

Dissertação submetida à Universidade Federal de Santa Catarina para obtenção do grau de Mestre em Ciência da Computação

Prof. Paulo José de Freitas Filho, Dr.

Orientador

Florianópolis

1997

DESENVOLVIMENTO DE TEMPLATES PARA MODELAGEM, SIMULAÇÃO E AVALIAÇÃO DE DESEMPENHO EM COMPUTADORES COM ARQUITETURA PARALELA

Adhemar Maria do Valle Filho

ESTA DISSERTAÇÃO FOI JULGADA ADEQUADA PARA OBTENÇÃO DO TÍTULO DE

MESTRE EM CIÊNCIA DA COMPUTAÇÃO

ESPECIALIDADE ARQUITETURA DE COMPUTADORES E APROVADA EM SUA FORMA FINAL PELO PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO.

Prof. Murilo Silva de Camargo, Dr
Coordenador do Curso

BANCA EXAMINADORA:

Prof. Paulo José de Freitas Filho, Dr.
Orientador

Prof. Paulo José da Silva Borges, Dr.

Prof. Thadeu Botteri Corso, M.Sc.

Agradecimentos

Esta dissertação tornou-se possível a partir do auxílio direto ou indireto de inúmeras pessoas. Cabe aqui o registro especial à figura do meu orientador que soube me conduzir sempre que me deparava frente a um problema sem solução abrindo novas perspectivas. Também seu papel foi importante em termos de recursos físicos, emprestando-me sua sala e equipamento de trabalho.

É importante ressaltar a atuação da coordenação, professores e secretárias do Curso de Pós-Graduação em Ciência da Computação através da organização e do fornecimento de várias disciplinas teóricas para o embasamento científico, passo fundamental para a consecução da dissertação.

Ao CNPQ e CAPES, órgãos financiadores de bolsas de pesquisa, agradeço por tornarem viável o presente trabalho e por acreditarem no empenho de professores e alunos.

Aos companheiros de laboratório e de sala de estudo, agradeço a colaboração tanto nas horas de trabalho quanto na organização do ambiente.

Minha gratidão aos meus pais e irmãs, um agradecimento especial a minha esposa que me incentivou e acompanhou-me nesta caminhada.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas

- ATM (*Assynchronous Transfer Mode*) - Modo de transmissão assíncrono;
- CISC (*Complex Instruction Set Computing*) - Conjunto de instruções complexas;
- COMA (*Cache Only Memory Access*) - Memória local transformada em cache;
- GUI (*Graphical User Interface*) - Interface voltada ao usuário;
- I/O (*Input/Output*) - Entrada/Saída;
- MIMD (*Multiple Instruction Multiple Data)* - Instruções múltiplas e dados múltiplos;
- MISD (*Multiple Instruction Single Data)* - Instruções múltiplas, dados simples;
- NORMA(*Non Remote Memory Access*) - Acesso a memória não remota;
- NUMA (*Non Uniform Memory Access*) - Acesso a memória não uniforme;
- RAM (*Random Access Memory*) - Memória de acesso direto;
- ROM (*Read Only Memory*) - Memória apenas de leitura;
- RISC (*Reduced Instruction Set Computerisation*) - Conjunto de instruções computacionais de tamanho reduzido;
- SIMD (*Single Instruction Multiple Data)* - Instruções simples, dados múltiplos;
- SISD (*Single Instruction Single Data)* - Instruções simples, dados simples;

- UMA (*Uniform Memory Access*) - Acesso à memória uniforme;
- VLSIW (*Very Large Scale Integration*) - Circuitos Integrados em muito-larga escala;
- SISD (*Single Instruction Single Data)* - Instruções simples dados simples;
- SIMD (*Single Instruction Multiple Data)* - Instruções simples, dados múltiplos;

Resumo

Apresenta-se neste trabalho o desenvolvimento e implementação de uma ferramenta para a modelagem e simulação de sistemas computacionais, cujo objetivo é facilitar e reduzir o tempo associado a esta etapa, em projetos de avaliação de desempenho, com a redução do tempo entre o projeto e sua implementação. Mais especificamente, a ferramenta destina-se a modelagem de sistemas que comportam uma arquitetura de processamento paralelo em configuração com linhas de interrupção e barramento de serviço. Ela reúne uma linguagem de simulação, possui uma interface amigável com alta flexibilidade e atende às características específicas daqueles sistemas. Estudos de sistemas complexos, como os que envolvem o projeto de arquitetura de computadores, necessitam de ferramentas para simulação e análise de desempenho das inúmeras propostas que se apresentam. A ferramenta aqui proposta fornece assistência ao projetista permitindo a análise do sistema em todas as etapas de modelagem e avaliação de performance do projeto, desde a coleta de dados até a apresentação e análise dos resultados. É oportuno o desenvolvimento deste tipo de pesquisa, uma vez que, a grande maioria das ferramentas existentes no mercado, não conseguem reunir as características que o usuário procura para encaminhar soluções aos inúmeros problemas que surgem ao longo do projeto. O trabalho foi desenvolvido a partir de conhecimentos nas áreas de arquitetura paralela, simulação e modelagem de sistemas juntamente com o projeto Nó // (leia-se nó paralelo) desenvolvido na Universidade Federal de Santa Catarina.

Abstract

This work presents the development and implementation of a tool for modeling and simulation of computer systems. Its main purpose is facilitate and abbreviate the time devoted to this step, reducing the gap between design and implementation. Specifically, the tool is suitable to model systems that involve parallel processing architecture with interrupt requests and bus service. The tool includes a simulation language and a GUI (Graphical User Interface) with high flexibility that is tailored for design those systems. The tool provides a support in every stage of modeling and performance evaluation, from getting data to the analysis and presentation of results. This is a convenient research development because of the majority of the existing tools in the market do not provide some characteristics that users are looking for to improve solutions to their problems. This work has been developed as a part of the research project called Parallel Node Project in Federal University of Santa Catarina. Parallel architecture processing, systems modeling and simulation are the main knowledge subjects that contributed to this research.

CAPÍTULO 1

1. Introdução

Trabalhos que envolvem construções de sistemas com alto grau de complexidade necessitam de ferramentas que auxiliem a análise para buscar a máxima performance e ao mesmo tempo permitam grande flexibilidade em seu uso. Este trabalho apresenta o desenvolvimento de uma ferramenta destinada à modelagem e simulação de sistemas com estrutura de processamento paralelo.

O projeto Nó // (leia-se nó paralelo) é um trabalho de pesquisa no qual associa grupos nas áreas de Arquitetura de Computadores, Sistemas Operacionais e Linguagens de Programação na Universidade Federal de Santa Catarina, que busca desenvolver um ambiente completo para a programação paralela [COR93]. O projeto está em fase adiantada, porém a máquina real ainda está em fase de construção. Qual arquitetura deve ser adotada, ou qual velocidade deve ter o barramento, são perguntas que surgem nesta fase de projeto. Para responder estas questões e auxiliar na tomada de decisões dispõe-se das ferramentas para a modelagem e simulação. Os produtos existentes não estão dirigidos a fornecer ao usuário uma interface com nomenclaturas do seu dia a dia e não permitem flexibilidade para a criação destas modelagens e simulações em um curto espaço de tempo.

A área de sistemas computacionais vem adquirindo uma maior complexidade a cada nova geração de máquinas. Para a análise destas novas máquinas surgem sistemas matemáticos

complexos que necessitam de ferramentas para seu auxílio. Na procura de soluções para a avaliação do problema, decisões de projeto e análise destes sistemas complexos, o analista encontra como alternativa vários caminhos, tais como a modelagem e simulação. Nestas ferramentas o projetista depara-se com as características antagônicas relativas à facilidade & flexibilidade. Com o propósito de construção de um modelo voltado para a análise de performance de sistemas, as dificuldades que um analista encontra são as mais variadas, entre elas estão o desconhecimento de uma linguagem específica de simulação, a falta de flexibilidade ao adotar uma linguagem de propósito específico, o ambiente de trabalho diferente do seu dia a dia e o tempo gasto até atingir um produto final para que possa realizar uma análise do modelo.

O objetivo deste trabalho é a construção de uma ferramenta para a modelagem de sistemas na área de arquitetura paralela. É uma ferramenta que procura minimizar as dificuldades encontradas por usuários, dá prioridade em apresentar um ambiente "natural" dos usuários potenciais e envolve os conceitos clássicos de modelagem e simulação. Com a utilização de Templates específicos gerados numa linguagem de simulação, o usuário não necessita de profundos conhecimentos de programação e os termos utilizados na interface fazem parte do seu ambiente de trabalho. A ferramenta apresenta uma interface amigável, que permite grande flexibilidade, exibindo ícones com figuras e termos utilizados normalmente em processamento paralelo. O objetivo da utilização desta ferramenta, além dos descritos anteriormente, é diminuir o intervalo que separa o projeto da modelagem do sistema.

Para a construção desta ferramenta foi escolhido um ambiente de modelagem, com flexibilidade e com uma interface gráfica voltada ao usuário (GUI - *Graphical User Interface*), que ao mesmo tempo permitisse o desenvolvimento de Templates específicos. Para tanto, desenvolveu-se uma metodologia que segue determinados passos:

- descrição do funcionamento do sistema;
- construção de um modelo referente ao sistema para o qual está voltada a confecção do Template;
- definição da interface para o usuário;
- validação.

A limitação desta ferramenta está no fato de se aplicar a arquiteturas que apresentem configuração com linhas de interrupção ou com barramento de serviço. Outra limitação importante é a possibilidade do aparecimento de novas tecnologias com características não incluídas na ferramenta construída.

Este trabalho está estruturado em 6 capítulos. No capítulo 2 apresenta-se uma visão geral dos estados da arte na área de arquitetura paralela no âmbito do projeto Nó //, assim como as ferramentas utilizadas para a sua avaliação. No capítulo 3 descreve-se a máquina Nó // com os elementos que fazem parte do projeto e no capítulo 4 conceitua-se "Modelagem" com exemplos utilizados na construção da ferramenta, que é o objeto deste trabalho. No capítulo 5 apresenta-se os conceitos de Template e a metodologia desenvolvida para a sua construção; no capítulo 6 examina-se os dois painéis construídos que fazem parte do Template e apresenta-se as interfaces de comunicação com o usuário, e no capítulo 7 expõe-se as conclusões e sugestões para futuros trabalhos a serem desenvolvidos sobre o tema. No anexo colocou-se a validação com tabelas e comparações.

CAPÍTULO 2

2. Revisão do Estado da Arte

2.1 Introdução

Neste capítulo apresenta-se uma visão geral dos estados atuais das artes nas áreas relacionadas com o trabalho realizado. Ele está dividido da seguinte forma: processamento paralelo, simulação, modelagem, projeto Nó // e as ferramentas utilizadas para sua avaliação.

A necessidade atual da maximização do desempenho das aplicações é devida, principalmente, à tendência dos sistemas no sentido ascendente de sofisticação e complexidade [MON95]. Para solucionar o problema nas áreas de cálculos, processamento de sons e de imagens, a computação necessita de mecanismos com elevado desempenho e alta confiabilidade, o que significa máquinas rápidas com grande poder de processamento [RIC96]. Com a disponibilização no mercado de processadores de baixo custo e da tecnologia de novas arquiteturas, idealizou-se a máquina paralela.

As máquinas paralelas foram projetadas de acordo com as necessidades em aplicações específicas. Foram desenvolvidas tecnologias de ponta, em várias configurações, dentro de Centros de Pesquisa em Universidades e Empresas. Para o acompanhamento e estudo destas novas tecnologias, desenvolveram-se também ferramentas especializadas, modelos de simulação que suportem o paralelismo, bem como técnicas de modelagem e avaliação de desempenho. Algumas destas ferramentas surgiram com o Projeto Nó // para auxílio ao estudo e pesquisa.

As ferramentas desenvolvidas constam de simuladores e modelos para avaliação, alteração e validação do projeto. Para a construção de um modelo de simulação, uma das maiores tarefas é a conversão da descrição do sistema em um programa para o computador [LAW94]. O projetista necessita da escolha de uma linguagem, que pode ser de propósito geral ou de propósito específico para simulação. As linguagens e softwares existentes no mercado que podem ser aplicados em computação paralela não possuem uma interface com termos que sejam ambientados com a área. O tempo aplicado pelo projetista em aprender uma linguagem pode ser reduzido com a utilização de uma interface com termos-padrão e de fácil manuseio.

2.2 Processamento paralelo

Arquiteturas paralelas são caracterizadas pela presença de múltiplos processadores que cooperam entre si na execução de uma única aplicação [DUN90]. A busca do alto desempenho para atender à crescente demanda e processamento motivou o aparecimento de vários modelos de arquiteturas paralelas.

Diversas taxonomias foram propostas para abranger as diferenças de projeto dos computadores que possuem paralelismo. Gordon Bell, em 1994, atualizou a classificação apresentada por Flynn em 1972, apresentando novas variações e modelos surgidos até então (Tabela 2-1 e Tabela 2.2).

Tipo de Fluxo de Instruções	Tipo de fluxo de dados	Tipo de Processamento
único	único	CISC
		RISC
		Superescalar e VLSIW RISC
		Processadores de sinais digitais
		"*Multi-threaded*"
	múltiplos	Vetoriais
		SIMD

Tabela 2-1 - Taxonomia para arquiteturas paralelas [BEL94]

Tipo de Fluxos de Instrução	Fluxo de dados	Tipo de acesso à memória	Armazenamento
Múltiplos	multiprocessador (memória compartilhada)	acesso a memória não uniforme	somente *cache* (COMA)
			"*memory coherent*"
			nenhum cache
		acesso a memória uniforme	"*multi-threaded*"
			supercomputadores e "*mainframes*"
			múltiplos barramentos
	multicomputador (passagem de mensagem)	chaveamento distribuído e centralizado	granularidade fina homogênea
			granularidade média não homogênea
			estações interconectadas na rede
		rede distribuída	aglomerados
			estações em rede local
			estações em rede externa

Tabela 2-2 - Taxonomia para arquiteturas paralelas (continuação) [BEL94]

A máquina que apresenta fluxos de instruções únicos é caracterizada pela execução de um fluxo em cada processador. Abrange desde os computadores CISC até os superescalares, VLSIW RISC, com palavras de instruções bem longas e computadores "*multi-threaded*" onde múltiplos "*threads*" processam o mesmo fluxo de instruções. Os processadores vetoriais com dados múltiplos SIMD são os núcleos dos supercomputadores largamente utilizados em aplicações com uso de ponto flutuante e aplicações com transformações gráficas.

O modelo que apresenta fluxos de instruções múltiplos denominados "MIMD" é composto de instruções múltiplas sobre fluxos distintos de dados em seus vários processadores. São divididos em duas classes de acordo com a forma de comunicação entre os processadores: os multiprocessadores e os multicomputadores.

Os multiprocessadores que apresentam memória compartilhada dividem-se em duas categorias: acesso à memória uniforme (UMA- *Uniform Memory Access*) e acesso à memória não uniforme (NUMA - *Non Uniform Memory Access*). Uma subclasse de NUMA possui cada nó de processamento com memória local transformada em cache (COMA- *Cache Only Memory Access*). Esta memória é caracterizada pelo seu acesso rápido. Outra subclasse, classificada como "coerência de memória", apresenta memória cache pequena, utilizada para acessos a dados presentes na memória do outro nó.

Os multicomputadores caracterizam-se por apresentar recursos de comunicação por passagem de mensagem. Um multicomputador é constituído por múltiplos nós associados através de uma rede de interconexão. Cada nó é um computador autônomo e possui um processador, memória e pode possuir discos ou periféricos de I/O (Figura 2-1).

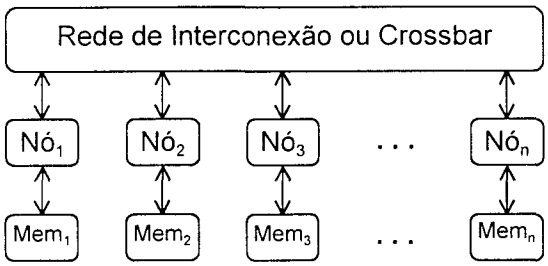


Figura 2-1- Estrutura geral de um multicomputador.

Em um multicomputador não existe compartilhamento de memória, e a interação entre os nós ocorre por meio da troca de mensagens pela rede de interconexão. Como não ocorre acesso a dados em memória remota, este tipo de arquitetura é também chamado de NORMA (*Non Remote Memory Access*).

Os multicomputadores podem ser agrupados em duas gerações, conforme o esquema de comunicação "entre os nós" utilizado: "armazena-e-repassa" ou "circuito chaveado". Os multicomputadores da primeira geração baseiam-se no esquema "armazena-e-repassa" para implementar um mecanismo de chaveamento, ou roteamento, de mensagens por *software*.

Na segunda geração de multicomputadores, é utilizada a técnica de "circuito chaveado", através de um mecanismo de chaveamento de mensagens por *hardware*. Em geral, os multicomputadores utilizam redes de interconexão estáticas, como a grelha ou o hipercubo [CAM95]. Alguns modelos adotam redes dinâmicas como o *crossbar* para estabelecer a comunicação entre os processadores da máquina.

Um exemplo de multicomputador da primeira geração é o Intel iPSC/1 [HWA93]. Entre os multicomputadores da segunda geração destacam-se o nCUBE/2, o Intel Paragon XP/S e o SuperNode1000 da Parsys [QUI94].

É importante observar que sistemas compostos por "*Workstations*", interconectadas por redes de alta velocidade como o ATM (*Assynchronous Transfer Mode*) podem ser classificados como computadores escaláveis e podem apresentar um custo de comunicação da ordem dos multicomputadores [MID96].

2.3 Ferramentas para Avaliação

As técnicas de avaliação de desempenho podem ser por modelagem analítica (teoria das filas), por simulação e por coleta de medidas. Para cada caso estudado, um conjunto de critérios de *performance* ou de medidas deve ser escolhido. Uma maneira de preparar este conjunto é fazer uma lista dos serviços que o sistema oferece. Critérios de como realizar as tarefas corretamente e tempo gasto para realizá-las, recursos utilizados para realização do serviço, podem ser considerados. Muitos projetistas que desenvolvem modelos analíticos consolidam seu projeto usando simulações e medições [JAI91].

Para determinar qual a técnica a ser utilizada, levantaram-se alguns pontos:

- a técnica por modelagem analítica não deverá envolver sistemas muito complexos devido à dificuldade dos cálculos, como, por exemplo, em teoria das filas.
- a técnica por coleta de medidas necessita da máquina real ou pelo menos de um protótipo.
- a técnica por simulação utiliza máquinas com alto poder de processamento e softwares adequados.

As técnicas para a modelagem são usadas para decidir o nível apropriado de detalhamento do modelo, tais como: definir no início do estudo as medidas de desempenho para avaliação, delinear todas as informações e dados no histórico do modelo e comparar as medidas de desempenho com uma avaliação anterior (se existir) [LAW94] [STR94].

Simulação é o processo de projetar o modelo de um sistema real e conduzir experimentos com ele no intuito de compreender seu comportamento e avaliar as estratégias para a operação do sistema. A simulação deve abranger a construção do modelo bem como a sua experimentação. É uma metodologia que procura descrever o comportamento do sistema, construir teorias ou hipóteses que tracem o comportamento observado e finalmente usá-lo para fazer previsões futuras, para verificar os efeitos produzidos por trocas no sistema ou no seu método de operação.

2.3.1 Aplicação da Simulação

A simulação é propícia em cada estudo de pesquisa na utilização e uso de técnicas de pesquisa operacional, devido à sua grande versatilidade, poder e flexibilidade. Quase todos os tipos de sistemas já foram ou podem ser simulados. Pode-se citar alguns exemplos:

⇒ Sistemas computacionais: componentes de hardware, sistemas de software, redes de computadores, gerenciamento e estruturação de dados, processamento de informação.

⇒ Manufatura: linhas de montagem, produção automatizada, armazenagem automatizada, sistemas de controle de inventário, estudos de manutenção e integridade.

⇒ Negócios: análises de ações e mercadorias, política de preços, estratégias de mercado, estudos para aquisição, análise de fluxo de caixa, previsão.

⇒ Governo: táticas militares, resgate médico, proteção contra incêndios, serviços policiais, desenhos de estradas e controle de tráfego.

⇒ Ecologia e ambiente: poluição da água e purificação, previsão do tempo, análise de terremotos e tempestades, sistemas de energia solar e escoamento de produção.

2.3.2 Vantagens e Desvantagens da Simulação

As vantagens da simulação começam pelo fato de que seu conceito é facilmente compreendido. Isto significa que ao apresentá-lo para um gerente junto com um modelo analítico, ele certamente irá preferir o modelo simulável.

Pontos favoráveis à simulação:

⇒ Pode-se explorar novos procedimentos, regras de decisões, estruturas organizacionais, fluxos de informações.

⇒ Novos projetos de hardware, *layouts* físicos, sistemas de transportes e programas de software podem ser testados sem comprometer recursos para aquisição e ou implementação.

⇒ Hipóteses sobre como ou porque certo fenômeno ocorre, se pode ser testado e se isto é viável ou não.

⇒ A execução da simulação pode ser controlada. Muitas vezes é necessário que o sistema execute várias operações rapidamente ou lentamente para observar determinados detalhes.

⇒ Pontos de estrangulamento podem ser constatados no fluxo de informações ou de dados.

⇒ Um estudo de simulação consegue fornecer importantes evidências de como o sistema funciona.

Desvantagens:

⇒ A construção de modelos requer pessoal especializado e a qualidade da análise depende muito da qualidade do modelo e do talento do projetista. A sua construção representa uma arte e o talento dos profissionais varia enormemente.

⇒ Os resultados de uma simulação, às vezes, são difíceis de ser interpretados. Quando colocam-se variáveis randômicas em um sistema real, é complicado determinar se uma observação é relacionada a uma combinação interna do sistema ou é efeito da casualidade randômica.

⇒ A análise por simulação pode consumir muito tempo e possuir um custo elevado. Uma análise pode não ser viável quando o tempo e os recursos são limitados. Por isso, para realizar um cálculo rápido, pode ser preferível, às vezes, utilizar métodos analíticos de modo manual.

2.3.3 O Processo de Simulação

O processo de simulação envolve técnicas para resolver problemas e prática em engenharia de software. Alguns pontos importantes são apresentados, os quais auxiliam a construção de um modelo de simulação e que, após a análise dos resultados, podem ajudar a decisão final referente a modificações e melhorias do desempenho do sistema:

⇒ Definir claramente os objetivos e planejar os recursos, como por exemplo, computadores e seus periféricos.

⇒ Definição de como o sistema opera.

⇒ Formulação conceitual do modelo, através de diagrama de blocos.

⇒ Projeto experimental com medidas, fatores a serem variados, dados necessários para o modelo, formato e quantidade.

⇒ Preparação para a entrada dos dados.

⇒ Formular o modelo em uma linguagem apropriada de simulação.

⇒ Validação e verificação - confirmar que o modelo opera da maneira que o analista previu e que os dados fornecidos representam fielmente a saída do modelo real.

⇒ Projeto experimental final - projetar um experimento que irá produzir a informação desejada e determinar quantas vezes o teste deverá rodar especificamente no modelo projetado.

⇒ Executar a simulação para gerar os dados desejados e efetuar uma análise mais discriminativa.

⇒ Interpretação e análise - traçar inferências a partir dos dados gerados pela simulação.

⇒ Implementação e documentação - colocar os resultados em prática, coletando os resultados e documentando-os.

2.3.4 Linguagens de Simulação

Uma das maiores tarefas na construção de um modelo de simulação é a conversão da descrição do sistema em um programa para o computador [LAW94]. Para construir um modelo de um sistema e posteriormente simulá-lo, necessita-se de uma linguagem de simulação. Esta linguagem deve oferecer vantagens como um ambiente flexível e que permita a facilidade de utilização para a área em que ela será aplicada.

O desenvolvimento de linguagens de simulação pode ser dividido em vários períodos [NAN84] [NAN93]. Por volta dos anos 60, todas as simulações eram escritas em uma linguagem de uso geral, muito difundida na época, o FORTRAN. Nos anos seguintes até meados de 65, apareceram linguagens para uso específico em simulação, como por exemplo GPSS, etc. Até os anos 70, aperfeiçoaram-se estas linguagens com revisões, novas versões e implementações. A evolução continuou e nos anos 80 surgem linguagens combinadas para a modelagem discreta e contínua. Ao mesmo tempo iniciam-se estudos para a mudança de critérios até então aplicados nas linguagens, visando facilitar o desenvolvimento de modelos e sua execução. Enquanto alguns autores se preocupavam com métodos potenciais para fácil programação [HEI74] [MAT74], outros procuravam dar-lhe uma abordagem mais formal [ZEI76] [KLE77] [NAN77]. A ênfase atual é prover uma interface amigável, com fácil interação entre o usuário e o programa, procurando simplificar a apresentação de resultados.

A taxonomia em linguagens de simulação pode ser resumida em três divisões: linguagens de uso geral, linguagens de simulação de uso geral e linguagens de simulação de propósito específico.

2.3.4.1 Linguagens de uso geral

Um modelo de simulação pode ser escrito em linguagens de propósito geral, como por exemplo C, BASIC, FORTRAN, PASCAL e C++ [JAC84].

Algumas vantagens das linguagens de programação de uso geral:

- O usuário, em geral, possui domínio da linguagem;
- C e FORTRAN são encontrados em vários computadores;
- O custo para aquisição de uma linguagem de uso geral é comumente mais baixo;

Algumas desvantagens:

- Alguns usuários podem não dominar uma linguagem de programação;
- A programação de um modelo de simulação exige um estudo mais apurado do software, pois será direcionada a um caso específico;
- A documentação deve ser bem detalhada para permitir a utilização por outros usuários;

2.3.4.2 Linguagens de simulação de uso geral

Existe também a possibilidade de modelagem pelo uso de linguagens de simulação de propósito geral. Como exemplo pode-se citar: GPSS/H, SIMAN/Cinema V, SIMSCRIPT II.5. A grande vantagem de um produto voltado à área de simulação é que ele fornece uma série de funções necessárias à modelagem de qualquer tipo de sistema, como por exemplo, a geração de números e funções aleatórias. A desvantagem é que para o usuário compreender esta complexa linguagem, necessita dispender de um tempo que poderia ser utilizado para outros fins.

2.3.4.3 Linguagens de simulação de propósito específico

A ênfase atual em desenvolvimento de softwares para simulação volta-se para a facilidade de uso por parte do usuário com a criação de ambientes integrados entre uma linguagem poderosa e flexível com uma interface amigável.

São produtos direcionados a objetivos mais específicos, quais sejam a modelagem de sistemas de telecomunicações, redes de computadores, manufatura, etc. Cita-se a seguir alguns exemplos [LAW94].:

a) SIMFactory - para a área de manufatura (pouca flexibilidade);

b) CONECT, BONeS DESIGNER e COMNET III - pacotes de modelagem de redes (pouca flexibilidade);

c) ARENA MP$im (Manufacturing Process Simulation) - pacote de simulação específico para a área de Manufatura (maior flexibilidade);

Mais recentemente, um novo objetivo tem-se imposto: a procura por ambientes que apresentem tanto a facilidade de uso (via interface amigável) quanto uma maior flexibilidade.

Nos exemplos anteriores, os itens a) e b) possuem como desvantagem a falta de flexibilidade, o que não acontece com o item c). A linguagem de programação de propósito específico ARENA MP$im permite o uso de suas características direcionadas à area de manufatura, porém com acesso a todos os recursos básicos oferecidos pela poderosa linguagem SIMAN e pelos Templates padrões do ARENA.

Pode-se citar algumas vantagens de uma linguagem de caráter específico:

- Redução do tempo de programação e conseqüentemente do custo do projeto;
- Facilidade de uso para profissionais da área, pois utiliza palavras conhecidas sobre redes de comunicações e computadores;
- Fornece as características implícitas de que o modelo necessita.

Algumas desvantagens são:

- A linguagem de programação de propósito específico necessita de um estudo mais aprofundado;
- Um software de propósito específico não se encontra disponível em qualquer computador de analistas;

- O software de caráter específico apresenta um custo mais elevado em comparação com um software de propósito geral;

Neste capítulo foram apresentadas algumas noções sobre processamento paralelo, incluindo taxonomias e características de cada tipo, dando ênfase ao multicomputador. Foram citadas as técnicas existentes para avaliação de sistemas, ou seja, analítica, de simulação e de coleta de dados. Para cada uma delas, apresentou-se as suas vantagens e desvantagens.

Capítulo 3

3. O Multicomputador Nó //

O Multicomputador Nó // é o nome dado à máquina com arquitetura paralela em desenvolvimento no Curso de Pós Graduação em Ciência da Computação da Universidade Federal de Santa Catarina. Neste capítulo será feita uma descrição geral da máquina citando-se as duas configurações básicas. Serão apresentados os componentes do multicomputador e ao mesmo tempo o seu funcionamento básico.

3.1 Descrição do Multicomputador Nó //

O multicomputador Nó // é composto por um conjunto de nós processadores e um nó de controle interligados por um meio físico denominado *crossbar*. Cada nó possui uma placa de interface para comunicação com outros nós. As conexões entre os nós são estabelecidas pelo *crossbar* dinamicamente, conforme a demanda do programa paralelo em execução. O nó denominado nó de controle tem a função de receber as requisições de serviço dos demais nós e atendê-las atuando sobre o *crossbar*.

Foram propostos dois modelos de arquitetura para o multicomputador Nó //: um modelo baseado em um barramento comum e outro baseado em um sistema de interrupção.

3.1.1 O Modelo de Arquitetura Baseado em um Barramento Comum

Foi a primeira concepção do multicomputador Nó //, com um conjunto de nós de trabalho interligado por meio de um comutador de conexões e um barramento compartilhado. O comutador de conexões é manipulado durante o funcionamento normal da máquina pelo nó de controle. Este utiliza um barramento de serviço e um *link* de configuração para receber

indicações do sistema e definir dinamicamente a estrutura da rede comunicação. O modelo é apresentado na Figura 3-1.

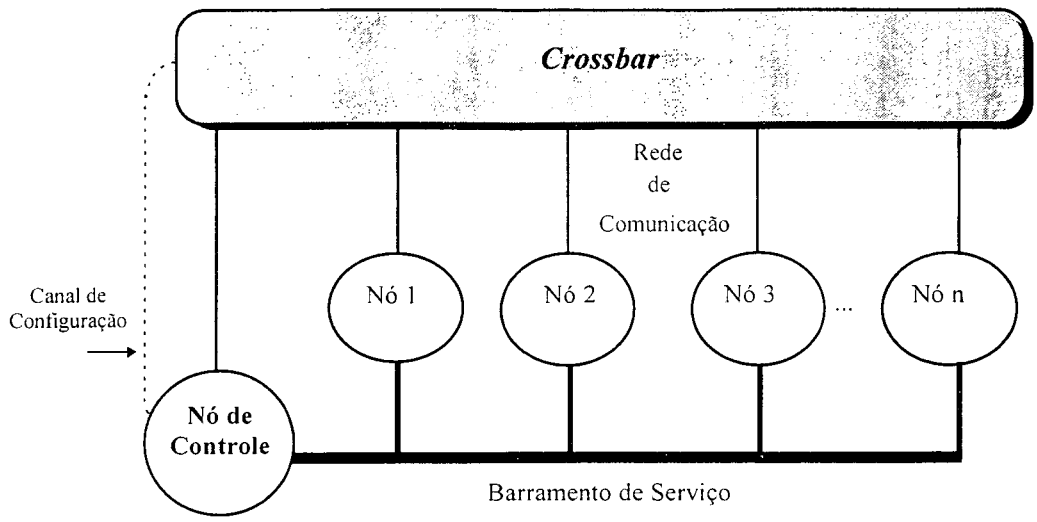


Figura 3-1 : Arquitetura com barramento de serviço

O nó de controle realiza uma busca seqüencial através do barramento de serviço para detectar algum pedido efetuado pelos nós de trabalho. Esta busca é feita através do envio de um comando ao nó inquirido. Se este não sinaliza algum serviço, o nó de controle questiona o próximo nó de trabalho. Caso o nó questionado necessite de um serviço, ele envia, então, uma requisição ao nó de controle. Após atender cada pedido, o nó de controle volta ao questionamento seqüencial aos nós de trabalho.

Se o pedido de serviço retornado pelo nó de trabalho for um pedido de conexão com outro nó, o nó de controle envia ao comutador de conexões a configuração para a conexão pedida. Se o nó solicitado já estiver conectado, o pedido é colocado em uma fila de espera e o nó de controle prossegue com o *polling*.

Esta arquitetura apresenta um inconveniente na execução do polling pelo nó de controle, pois se um nó de trabalho que foi questionado quiser fazer um pedido de serviço, terá que esperar todos os outros nós da seqüência serem questionados e atendidos, até chegar novamente a sua vez. Este problema é agravado quando se aumenta o número de nós.

3.1.2 O Modelo de Arquitetura Baseada em um Sistema de Interrupção

O modelo descrito a seguir é uma nova arquitetura com o objetivo de melhorar o mecanismo apresentado no item anterior. Ele é composto por linhas de interrupção substituindo o barramento de serviço. Estas linhas são exclusivas entre os nós de trabalho e o nó de controle (Figura 3-2), representadas por um par de linhas unidirecionais referidas por INTRtc e INTRct. A linha INTRtc é usada pelo nó de trabalho para interromper o nó de controle, enquanto que a linha INTRct é utilizada pelo nó de controle para interromper o nó de trabalho.

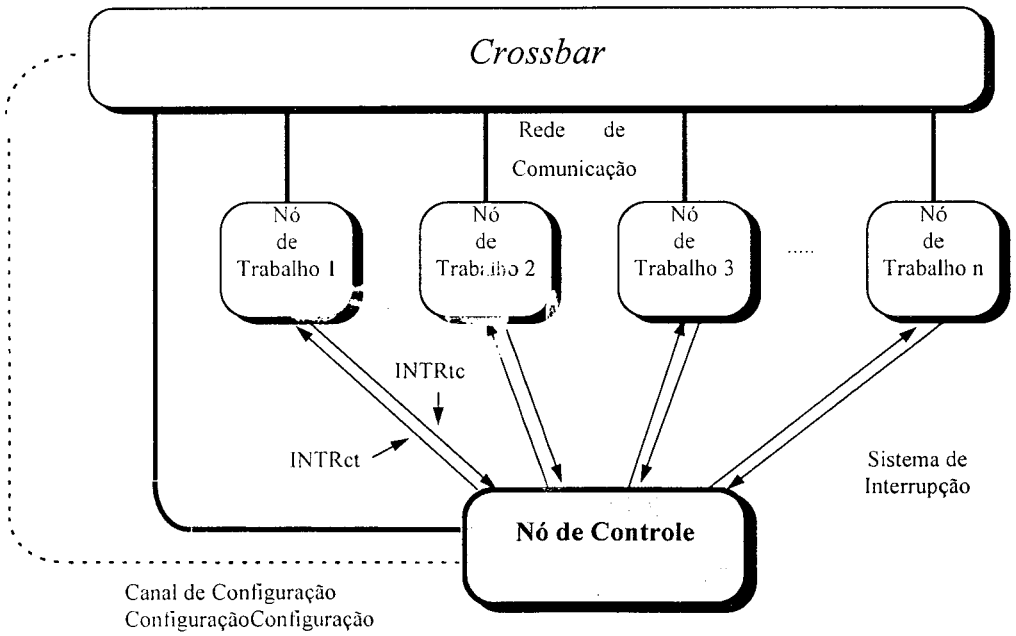


Figura 3-2 : A Arquitetura Baseada em um Sistema de Interrupção para o Multicomputador Nó//

As linhas de interrupção têm a função de indicar a ocorrência de um evento e não podem ser utilizadas para a transferência de dados. Quando o nó de trabalho necessita enviar uma requisição de serviço ao nó de controle, interrompe-o através da linha INTRtc. O nó de controle, conforme a prioridade, estabelece conexão via *crossbar* com o nó que gerou a interrupção. Estabelece a troca de mensagens de controle através da requisição de serviço e após a troca a conexão é desfeita. O nó de controle avaliará a requisição e, no momento de atendê-la, ativa a linha de interrupção INTRct do nó de trabalho requisitante, que vai permitir a realização do serviço requisitado.

As transferências de dados ou de mensagens de controle somente são efetuadas através da rede de comunicação. O atendimento a uma requisição não será seqüencial, tipo *polling*. Isso acarreta a diminuição no *overhead* associado à comunicação, pois depende apenas do atendimento da interrupção e do chaveamento do *crossbar*.

3.2 Componentes do Multicomputador Nó //

O multicomputador Nó // é subdividido em duas partes: a rede de comunicação e o sistema de interrupção. Neste item serão descritos os componentes do Multicomputador Nó// a seguir citados: nós de trabalho, nó de controle, comutador de conexões ou *crossbar*, canais de comunicação e as placas de interface.

3.2.1 Os Nós de Trabalho

O Multicomputador Nó // é constituído de nós com processadores i486, memória RAM privativa de 4Mb, memória ROM com o sistema operacional microcódigo, interface com a rede de comunicação e interface com o sistema de interrupção.

Cada nó de trabalho é conectado ao *crossbar* por canais ou vias de comunicação que formam a rede de comunicação.

A Figura 3-3 apresenta a estrutura do nó de trabalho:

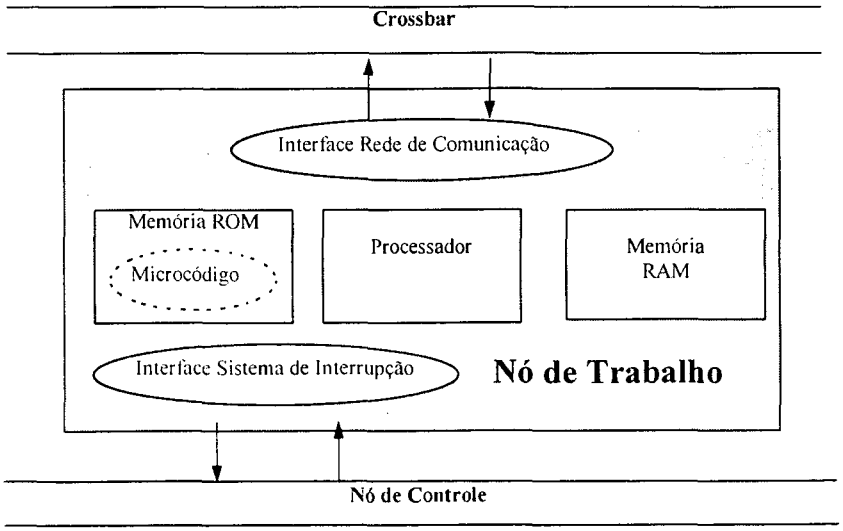


Figura 3-3 : Estrutura do nó de trabalho

3.2.2 O Nó de Controle

A rede de comunicação possui um gerenciador que controla o *crossbar* e os nós de trabalho. Este componente é o nó de controle e sua estrutura é apresentada na Figura 3-4.

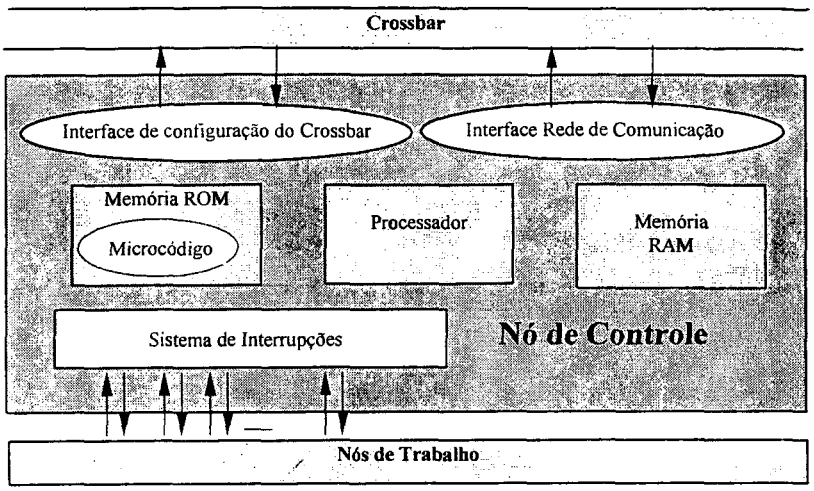


Figura 3-4 : Estrutura do nó de controle

3.2.3 O *Crossbar*

O dispositivo utilizado para comutar as conexões é denominado de *crossbar*. Na Figura 3-5 temos o diagrama de blocos do *crossbar*. Ele é constituído de um comutador de conexões programável projetado para fornecer um chaveamento completo entre 32 linhas de entrada e 32 linhas de saída. As entradas e saídas são identificadas por um número na faixa de 0 a 31.

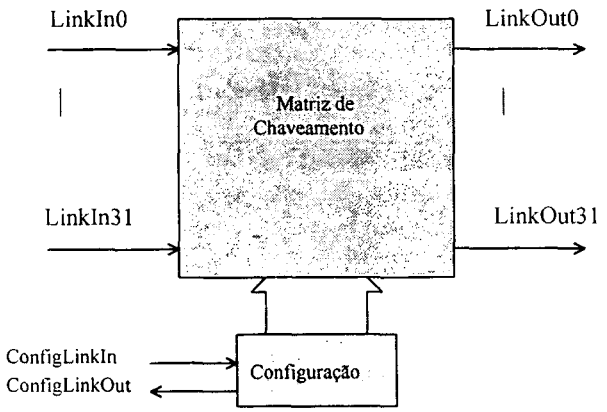


Figura 3-5: Diagrama de blocos básico do *crossbar*

O *crossbar* é programado através de um canal bidirecional serial denominado *link* de configuração As mensagens de configuração (*ConfigLink*) consistem de um, dois ou três *bytes*, e fazem comunicação com o sistema através dos canais de comunicação. O número de canais que interliga o nó de trabalho e o *crossbar* é implementado de acordo com o hardware, em número de 1 a 4.

3.2.4 Os Canais de Comunicação

Os canais de comunicação utilizados são ligações entre o *crossbar* e os nós processadores, que no caso são os nós de trabalho e o nó de controle. Os canais formam a rede de comunicação e estabelecem uma comunicação *full-duplex* com microprocessadores padrão e com outros sub-sistemas, realizando a conversão de um canal serial bidirecional em uma interface paralela de oito *bits*, e vice-versa.

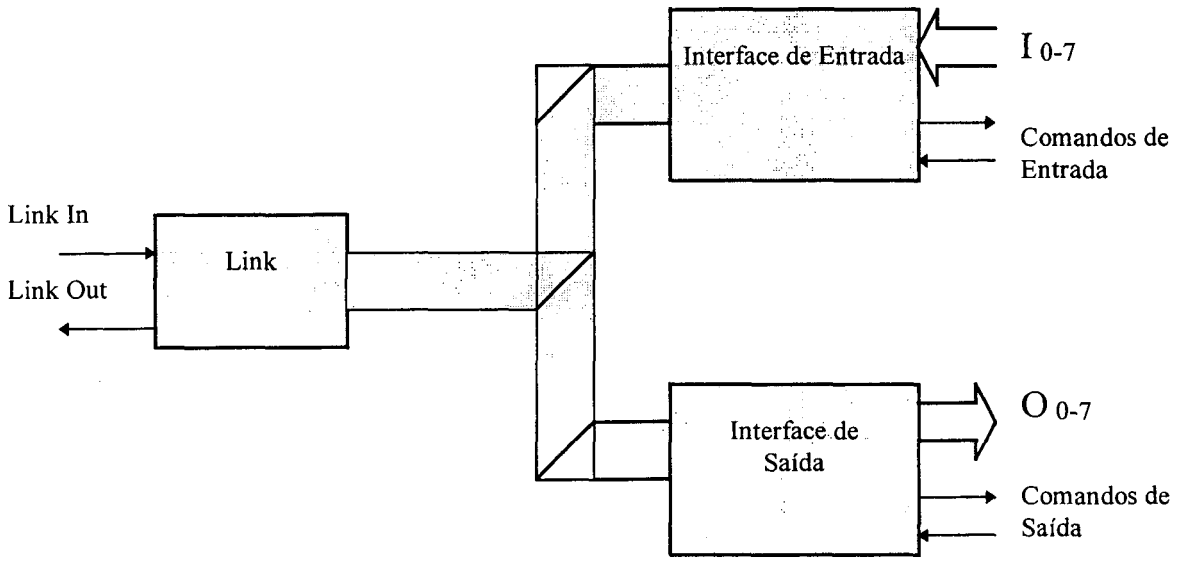


Figura 3-6: Diagrama de blocos dos Canais de Configuração

Na Figura 3-6 observa-se a interface de Entrada controlada por portas lógicas denominadas de comandos de entrada. Elas controlam o início e o término da comunicação. Da mesma maneira funciona a Interface de Saída. Um *byte* enviado por um *link* é reconhecido na entrada do mesmo *link*, assim cada linha pode transportar dados e informações de controle.

3.2.5 Placas de Interface

As interfaces da rede de comunicação, do sistema de interrupção e a de configuração do *crossbar* (pertencente apenas ao nó de controle) estão localizadas em uma placa compatível com o barramento PC-AT. A essência das interfaces de rede de comunicação e configuração do *crossbar* está no *link adaptor* utilizado, o qual fornece um *link* serial para transferência de dados em alta velocidade. A seguir serão apresentadas as estruturas das placas de interface para os nós de trabalho e de controle, que servem para implementar a arquitetura do Nó //.

3.2.5.1 Placa de Interface dos Nós de Trabalho

A interface da placa com o barramento PC-AT é responsável pela comunicação entre o microprocessador e as outras interfaces. A interface com a rede de comunicação é utilizada para a comunicação entre os nós de trabalho e também entre o nó de controle e os nós de trabalho. As comunicações se dão através de conexões estabelecidas pelo *crossbar*. A interface com o sistema de interrupção gera uma interrupção (INTRtc) que é enviada ao nó de controle para caracterizar a solicitação de um pedido, e detecta a chegada de uma interrupção gerada pelo nó de controle (INTRct) (Figura 3-7).

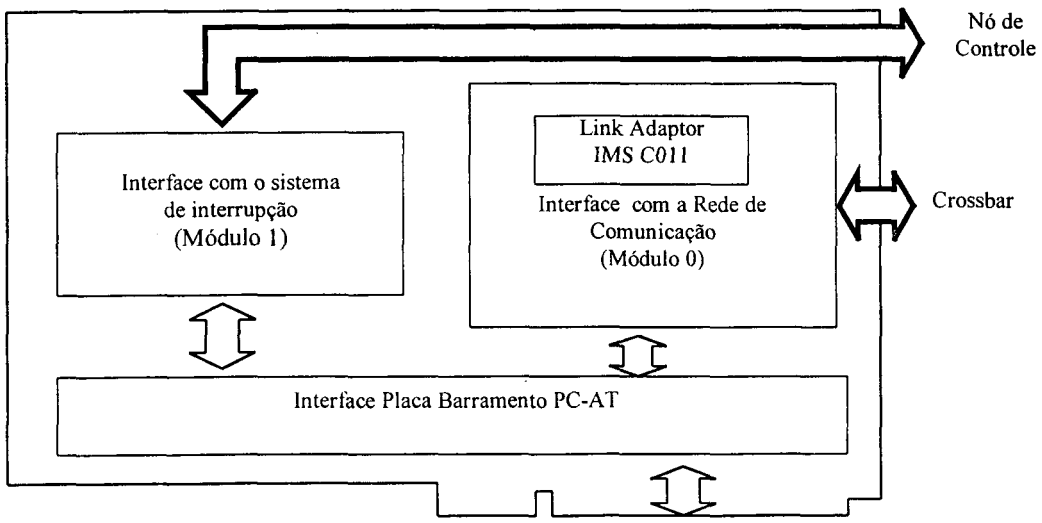


Figura 3-7 : Diagrama de Blocos da Placa de Interface do nó de trabalho

3.2.5.2 Placa de Interface do Nó de Controle

A placa de interface do nó de controle é semelhante à placa dos nós de trabalho, acrescentando a esta uma interface com o canal de configuração do *crossbar*. Ela é composta de três módulos e uma interface para a placa de barramento do computador.

A interface com a rede de comunicação (Módulo 0) é utilizada para a comunicação entre os nós de trabalho e entre o nó de controle e os nós de trabalho. A interface com o sistema de interrupção (Módulo 1) gera uma interrupção (INTRtc), que é enviada ao nó de controle para caracterizar a solicitação de um pedido, e detecta a chegada de uma interrupção gerada pelo nó de controle (INTRct). Por intermédio da interface denominada de Módulo 2, o nó de controle envia os sinais *ConfigLinkIn* e *ConfigLinkOut*, fornecendo as informações que configuram o *crossbar* (Figura 3-8).

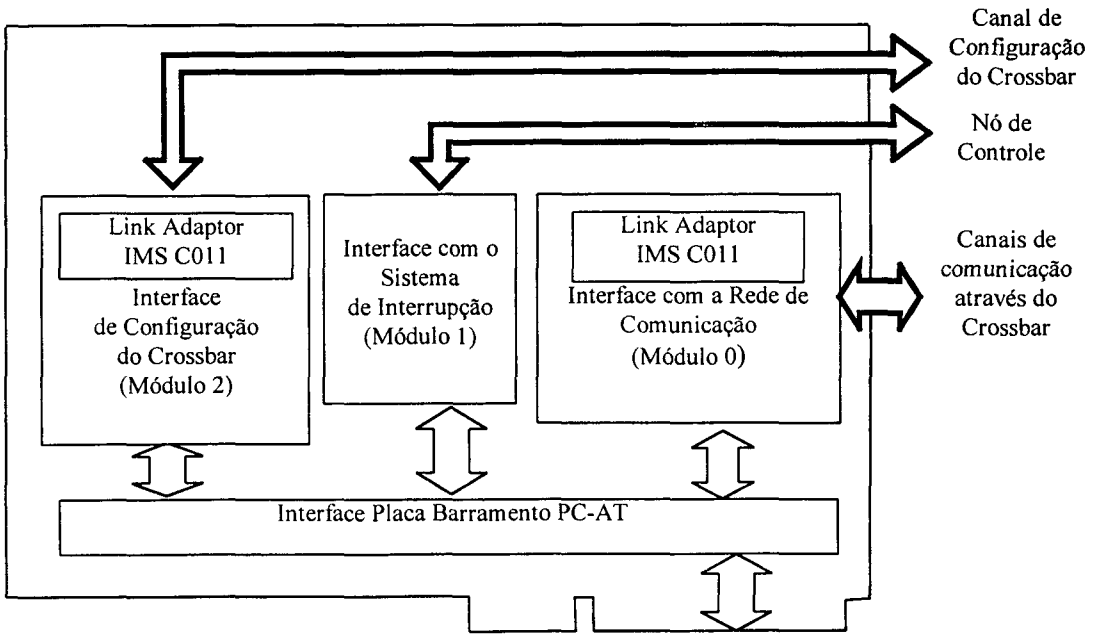


Figura 3-8 : Diagrama de Blocos da Placa de Interface do nó de controle

Neste capítulo foi detalhada a construção da máquina pertencente ao projeto Nó //. Foram apresentadas as duas arquiteturas paralelas e suas configurações com barramento de serviço e com linhas de interrupção, juntamente com a descrição do funcionamento de cada uma. Em seguida, os componentes da máquina paralela foram descritos sucintamente. Estes componentes farão parte do modelo que será elaborado para cada uma das arquiteturas. Com o intuito de elaborar o modelo, alguns conceitos sobre modelagem serão mostrados no capítulo seguinte.

CAPÍTULO 4

4. Modelagem

Neste capítulo apresenta-se inicialmente os conceitos de modelagem e seu uso, com o intuito de simulação, sendo seu ambiente exposto, com suas peculiaridades, objetivando iniciar a construção do modelo que representa a arquitetura paralela. O tópico abordado neste capítulo é parte da metodologia utilizada para a elaboração do Template.

4.1 Definição

Para conceituar modelagem, inicialmente será apresentado uma definição do que é um sistema. Este é identificado quando se isola parte de um ambiente; define-se então um sistema como sendo este mundo fechado no qual se separa os itens que são parte integral dele (que estão dentro do mesmo) e itens que podem afetá-lo externamente. São usados modelos para realizar a comunicação entre ambos. Utiliza-se para esta comunicação elementos, tais como, equações, gráficos e diagramas. O modelo reflete a metáfora ou a linguagem utilizada para falar sobre o sistema [FIS95].

Outra conceituação é aquela que engloba as suposições utilizadas para descrever um sistema ou processo do mundo real e suas abstrações, na tentativa de obter alguma compreensão do seu comportamento e torná-lo uma relação matemática ou lógica [LAW91].

Por uma modelagem entende-se a representação de um grupo de objetos ou idéias. Historicamente, esta tomou vários formatos, desde a comunicação através de desenhos em cavernas até complexos sistemas de equações matemáticas para o vôo de uma aeronave no espaço. O progresso e a história da ciência e da engenharia estão mais precisamente refletidos na evolução da habilidade humana em desenvolver e usar esses modelos [PEG95].

Utiliza-se modelos para aprender alguma coisa sobre um sistema real que não se pode observar diretamente, por motivos da sua inexistência ou pela dificuldade de manuseio. Um modelo cuidadosamente concebido pode abandonar a sua complexidade e abranger o estritamente necessário ao analista. O mesmo pode ser da forma mais variada, mas aquele que certamente é o mais importante no contexto deste trabalho é o que realiza a simulação.

4.2 Tipos de Modelos

Neste item serão apresentadas as classificações dos modelos de simulação. A taxonomia pode se dar através dos seguintes aspectos: quanto às aplicações, quanto ao seu comportamento em relação ao tempo, quanto à troca de estados dos recursos utilizados e quanto à presença de variáveis randômicas.

No que tange às aplicações, os modelos podem ser:

- modelos de simuladores de imagens, como por exemplo, simuladores de vôo
- simuladores simbólicos, nos quais as propriedades e características de um sistema real são representadas de uma forma matemática ou simbólica.

Uma outra classificação separa sistemas reais que estão livres da influência de

variáveis randômicas ou imprevisíveis no seu ambiente ou em seus próprios componentes:

- modelos de simulação determinísticos que ignoram as variáveis randômicas;
- modelos estocásticos, que consideram o componente randômico como parte do sistema.

Outra maneira de classificar os modelos de simulação está relacionada com o tempo:

- modelos dinâmicos, que descrevem o comportamento do sistema através do tempo;
- modelos estáticos, que retratam o comportamento do sistema em um ponto fixo do tempo.

Para citar mais uma classificação, tem-se a maneira como o modelo representa a troca de estados dentro do sistema:

- modelos discretos, que apresentam trocas de estados em pontos isolados no decorrer do tempo;
- modelos contínuos, onde a troca se dá continuamente ao longo do tempo;
- combinados, que alternam os dois tipos de variações ao longo do tempo, discretas e contínuas.

Os modelos tratados neste trabalho classificam-se como: simbólicos, estocásticos, dinâmicos e discretos. Simbólicos, porque são processados em computadores; estocásticos, pois utilizam a randomização; dinâmicos, porque descrevem o comportamento do sistema através do tempo e discretos, pois alternam estados de ocupação do recurso no decorrer do tempo.

4.3 Ambiente para Modelagem e Simulação

No âmbito do Laboratório da Pós Graduação em Ciência da Computação utiliza-se o ambiente de modelagem ARENA 2.1, o qual executa a linguagem de simulação SIMAN V.

Esse ambiente possui razoável flexibilidade de uso, permitindo que, num mesmo modelo, se utilize recursos de alto e baixo nível de agregação.

O ambiente ARENA permite a modelagem sem entrar em muitos detalhes a nível da linguagem SIMAN. Esta característica demonstra um alto nível de agregação. Já a linguagem de simulação SIMAN utiliza painéis da própria linguagem, que possuem baixo nível de agregação.

O Arena contém inúmeros recursos para modelagem e simulação, incluindo facilidades como: um ambiente de programação gráfica, recursos de animação, possibilidade de uso da linguagem de simulação SIMAN V e elaboração de relatórios com detalhes dos resultados.

SIMAN é uma poderosa linguagem de simulação para delinear sistemas discretos ou contínuos, permitindo também modelar uma combinação entre eles. Sistemas com alteração discreta podem ser representados usando tanto uma orientação voltada à interação de processos ou a eventos escalonados no tempo. Sistemas com alteração contínua são moldados algebricamente ou com equações diferenciais. A combinação destas orientações é utilizada para retratar sistemas ajustados.

A lógica utilizada em SIMAN é segmentada em duas partes, a parte do modelo e a parte do experimento. O modelo descreve os elementos físicos do sistema, tais quais: processadores, linhas de dados, circuitos lógicos, memórias, etc. O experimento é utilizado para especificar sob quais condições a simulação será executada, incluindo elementos como: condições iniciais, disponibilidade dos recursos, tipo de estatísticas a serem registradas e tempo da simulação.

Para apresentar o ambiente de modelagem são necessários alguns conceitos como como templates, painéis e módulos.

4.3.1 Componentes do Ambiente de Modelagem do ARENA

O ambiente de Modelagem é composto pelos seguintes elementos: área de trabalho, menu de ferramentas e menu de templates. Cada um destes elementos será apresentado e descrito.

Na Figura 4-1 mostra-se o ambiente de trabalho oferecido pelo ARENA, destacando-se os componentes mais importantes: menu de ferramentas, ícones, painéis, etc.

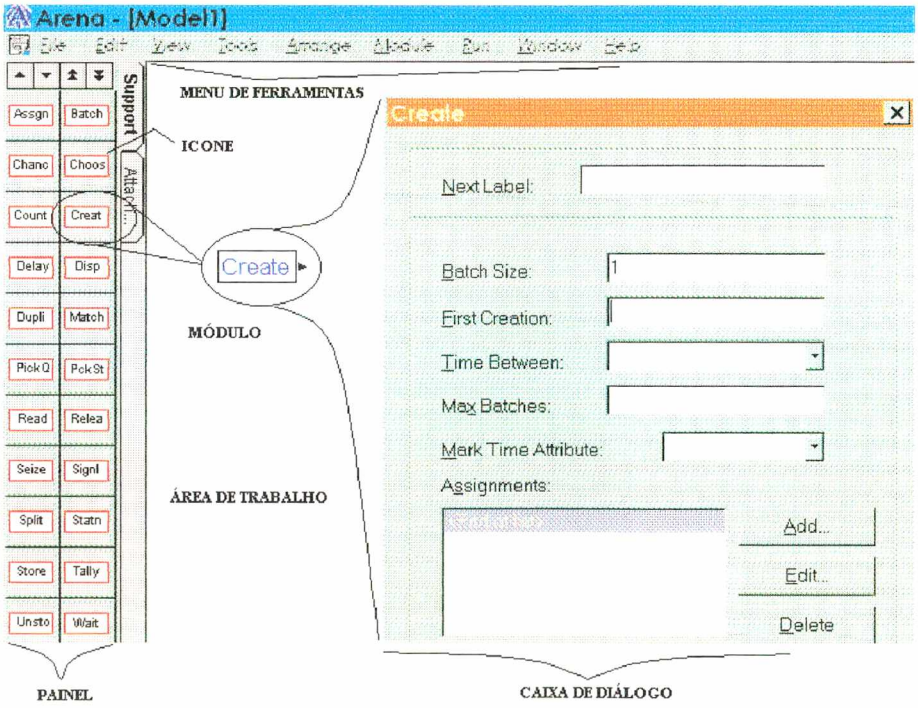


Figura 4-1- Ambiente de trabalho

Do lado esquerdo tem-se o local onde agrega-se os painéis. Eles fazem parte de um conjunto maior denominado template. Na ilustração aparece o painel selecionado, que é o Support, e faz parte do template do ambiente ARENA. Os painéis são compostos por módulos, destacando-se na figura o módulo CREATE e sua caixa de diálogo correspondente.

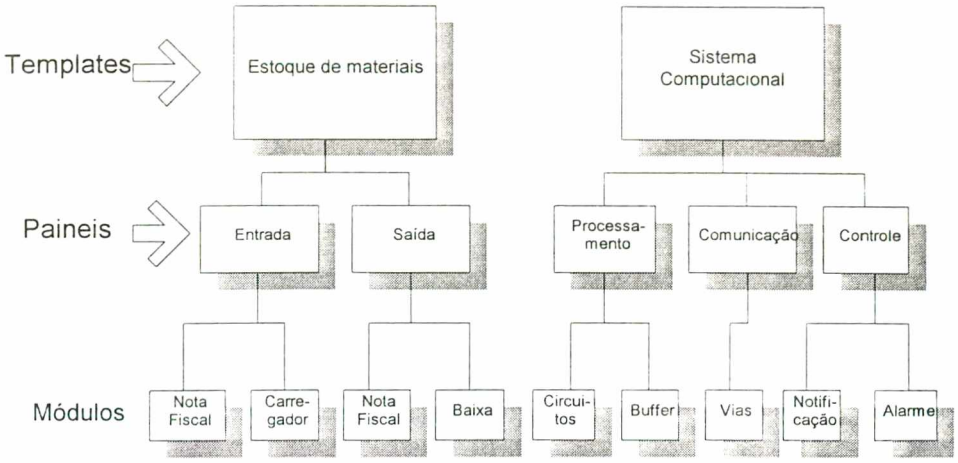


Figura 4-2- Ilustração representativa sobre templates e painéis

Para esclarecer a diferença entre templates e painéis, a Figura 4-2 mostra um diagrama esquematizando um Sistema Computacional e um Estoque de Materiais. Um template é como uma divisão principal e um painel seria uma sub-divisão. No exemplo, o "Estoque de Materiais" é dividido em "Entradas e Saídas", o que corresponde no caso do Sistema Computacional a "Processamento", "Comunicação" e "Controle". Dentro de cada sub-divisão ainda existe uma outra separação que corresponde aos módulos. Por exemplo, a entrada dentro do estoque é dividida em "Nota Fiscal" e "Carregador". Similarmente, no "Sistema Computacional" tem-se o "Processamento em Circuitos" e "*Buffers*".

Módulos são unidades básicas dos painéis e funcionam como comandos de uma linguagem de programação como por exemplo FORTRAN. Cada módulo possui sua caixa de diálogo onde o usuário coloca os dados no modelo. A facilidade oferecida por estes blocos é devida ao fato de que eles foram projetados sob a ótica da simulação em conjunto com a interface gráfica do ARENA.

Para construir um modelo, seleciona-se vários módulos deste mesmo painel ou pode-se escolher outro painel do mesmo conjunto de templates. Os blocos da linguagem SIMAN também podem ser utilizados na construção dessa modelagem. A flexibilidade é uma característica marcante neste ambiente de programação. A seqüência lógica desses módulos conectados em conjunto com os dados fornecidos permite a execução do modelo.

Os modelos são criados graficamente e executados interativamente, permitindo rodar passo a passo ou de uma forma rápida através de controles existentes no menu de interação (Figura 4-3), o qual se localiza na parte inferior da tela de trabalho. Estas facilidades na execução do programa permitem verificar cada seqüência executada e localizar algum procedimento incorreto.

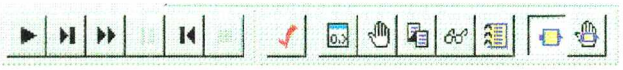


Figura 4-3- Menu de Interação com a simulação do modelo

O modelo pode ser feito com animação como é apresentado na Figura 4-4 com gráficos, ilustrações que representam recursos ocupados, relógios e contadores. No momento da execução pode-se selecionar com ou sem animação. Este recurso permite a verificação do comportamento do sistema passo a passo e também pode ser utilizado para detectar procedimentos indevidos.

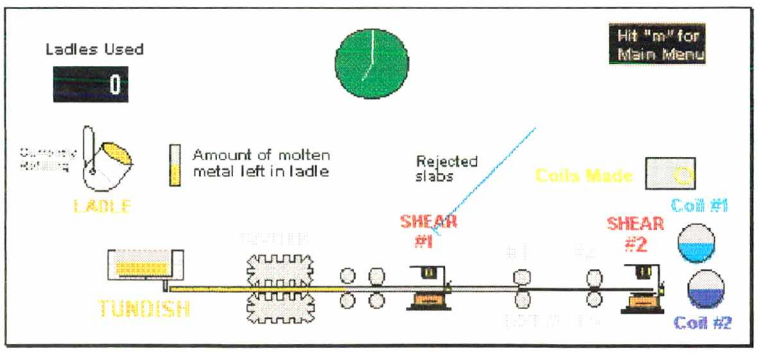


Figura 4-4- Área de trabalho com animação

4.4 Exemplo de Modelagem no ARENA 2.1

Como exemplo de desenvolvimento de modelagem, apresenta-se dois modelos que simulam a operação da máquina Nó Paralelo e que serão posteriormente apresentados ou reconstruídos utilizando a ferramenta do qual este trabalho é objeto.

Os dois modelos constituem-se de elementos básicos, utilizados em uma máquina paralela, cujos componentes principais são:

- nós processadores ou nós de trabalho;
- nó de controle central;
- comutador de conexões tipo *crossbar*;
- barramento de serviço.

4.4.1 Modelo com Linhas de Interrupção

O modelo construído utiliza a interface padrão de modelagem do ARENA. Esta interface padrão é orientada a processos que são ativados por entidades. As entidades são objetos que se movem ao longo do sistema, causando mudanças de estado, realizando uma seqüência de operações, como por exemplo, a alocação de um recurso. Este recurso possui um *status* que pode livre ou ocupado.

Nesta modelagem, as entidades representam as mensagens e os recursos representam os processadores. Os recursos serão alocados dinamicamente ao longo do tempo através de entidades que circulam pelo modelo e agregam determinadas características ou atributos para simularem o funcionamento real do sistema. Tais características, dependendo do sistema a ser modelado, podem ser alteradas a cada simulação.

A modelagem de um nó de trabalho foi construída utilizando-se 19 módulos padrões do ARENA, como pode ser observada na Figura 4-5.

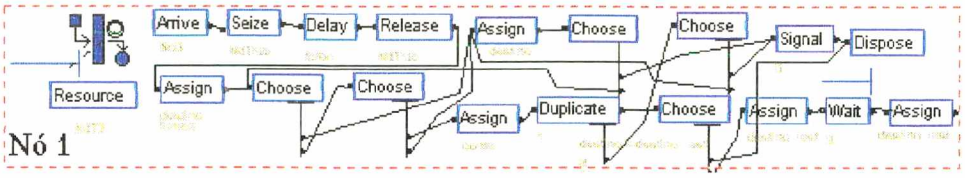


Figura 4-5 - Conjunto de módulos que compõe o nó de trabalho 1

Estes módulos possuem a função básica de gerar as mensagens iniciais e processar parte das variáveis. O retângulo azul, representando o recurso, compõe o nó de trabalho 1. Este modelo foi projetado para quatro nós de trabalho, significando, portanto, que, estes 19 módulos serão repetidos mais quatro vezes a fim de se compor o modelo completo.

Em relação aos módulos que compõem o Controlador de nós ou Conexões, que é o responsável no Nó// por comandar o comutador de conexões através do canal de configuração, são apresentados na Figura 4-6. Para compor o nó de controle foram utilizados 8 módulos básicos do ARENA.

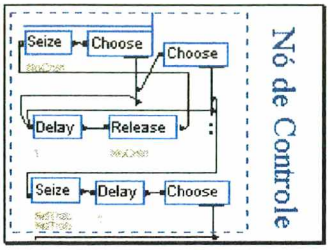


Figura 4-6- Módulos que compõem o nó de controle

O conjunto de módulos apresentado a seguir define o *crossbar* ou comutador de conexões. Ele representa o meio físico onde se estabelecem as comunicações entre os nós. Na Figura 4-7 estão representados os 22 módulos que executam a função do *crossbar*.

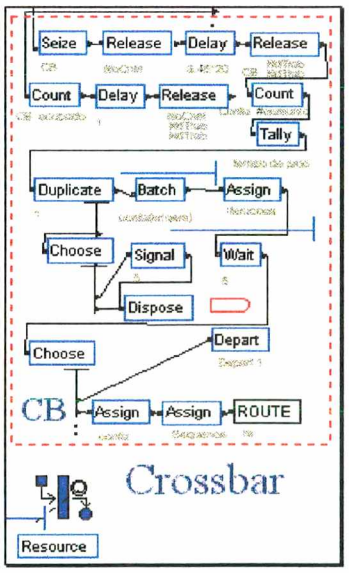


Figura 4-7- Módulos que fazem parte do *crossbar*

Na página seguinte apresenta-se a Figura 4-8 com a visão geral do modelo relativo à arquitetura paralela com linhas de interrupção.

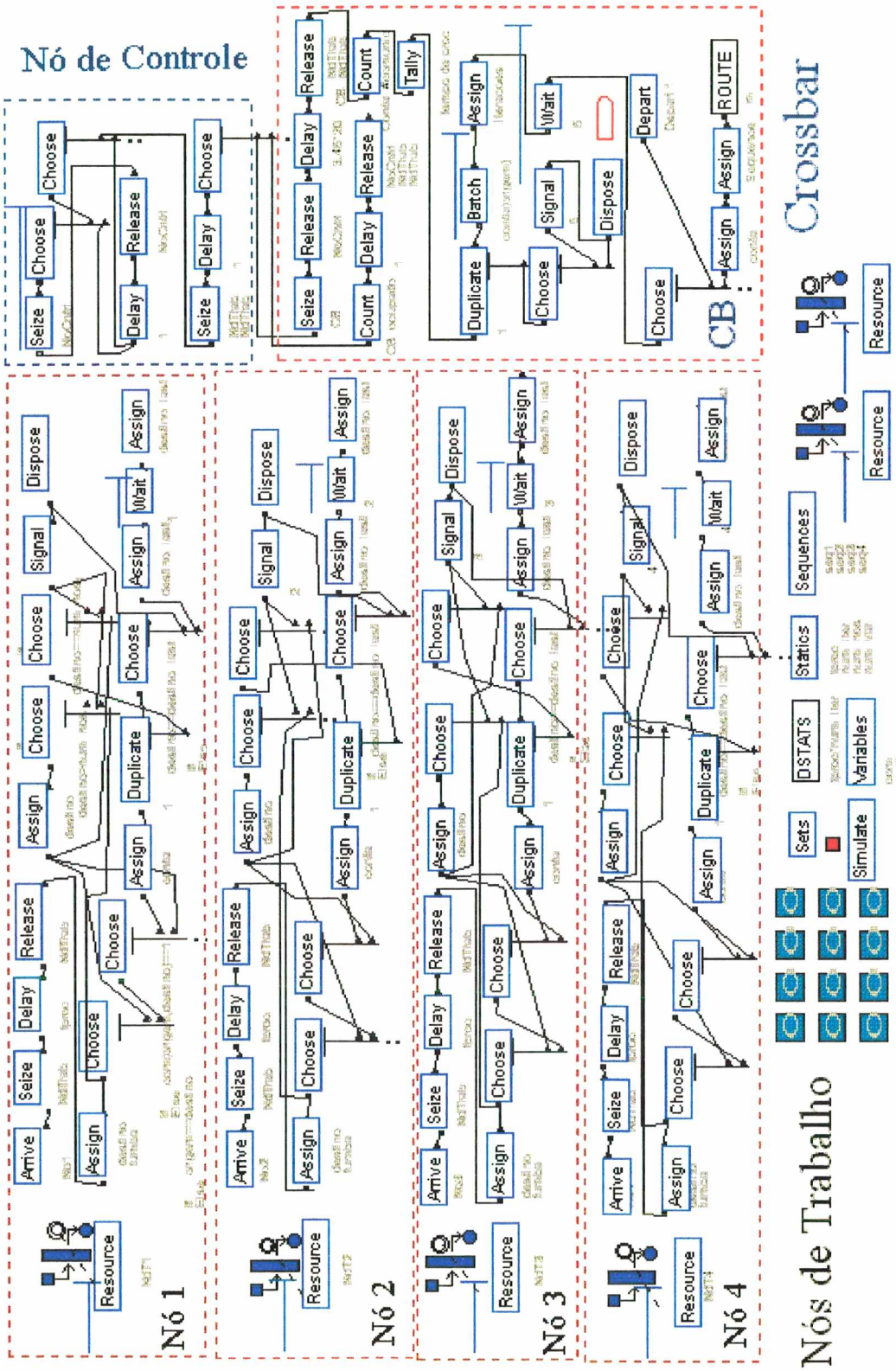


Figura 4-8: Visão geral do Modelo que simula o Nó// na arquitetura com Linhas de Interrupção

4.4.2 Modelo com barramento de serviço

Os módulos que compõem o barramento de serviço são semelhantes ao modelo com linhas de interrupção. A diferença no funcionamento dos dois modelos, e portanto refletindo na sua lógica de modelagem, está relacionada com os módulos que tratam do processamento no nó de controle. No modelo com interrupção, a mensagem chega em uma fila existente no nó de controle. O nó de controle, ao perceber que um nó de trabalho deseja efetuar uma comunicação, configura o *crossbar* e verifica o destino da comunicação, efetuando, a seguir, a conexão desejada. No caso do barramento de serviço, as mensagens aguardam nos nós de trabalho, e um dispositivo simulando o *polling* efetua uma varredura na fila de cada processador. O barramento de serviço é concebido apenas para troca de mensagens curtas e pré-estabelecidas entre um nó qualquer e o nó de controle. Não acontecem colisões no acesso ao barramento de serviço porque é o nó de controle que determina com quem será realizada a conexão para verificar o destino da comunicação. O número de iterações também pode ser configurado e representa o número de vezes que o processo deverá ser repetido de acordo com o sistema real.

O conjunto de módulos mostrado a seguir representa as diferenças lógicas de controle que foram apresentadas no parágrafo anterior.

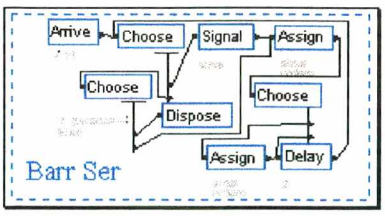


Figura 4-9- Conjunto de módulos que representa o barramento de serviço

Na Figura 4-10 da página seguinte, apresenta-se a visão geral do modelo que contempla a arquitetura com barramento de serviço.

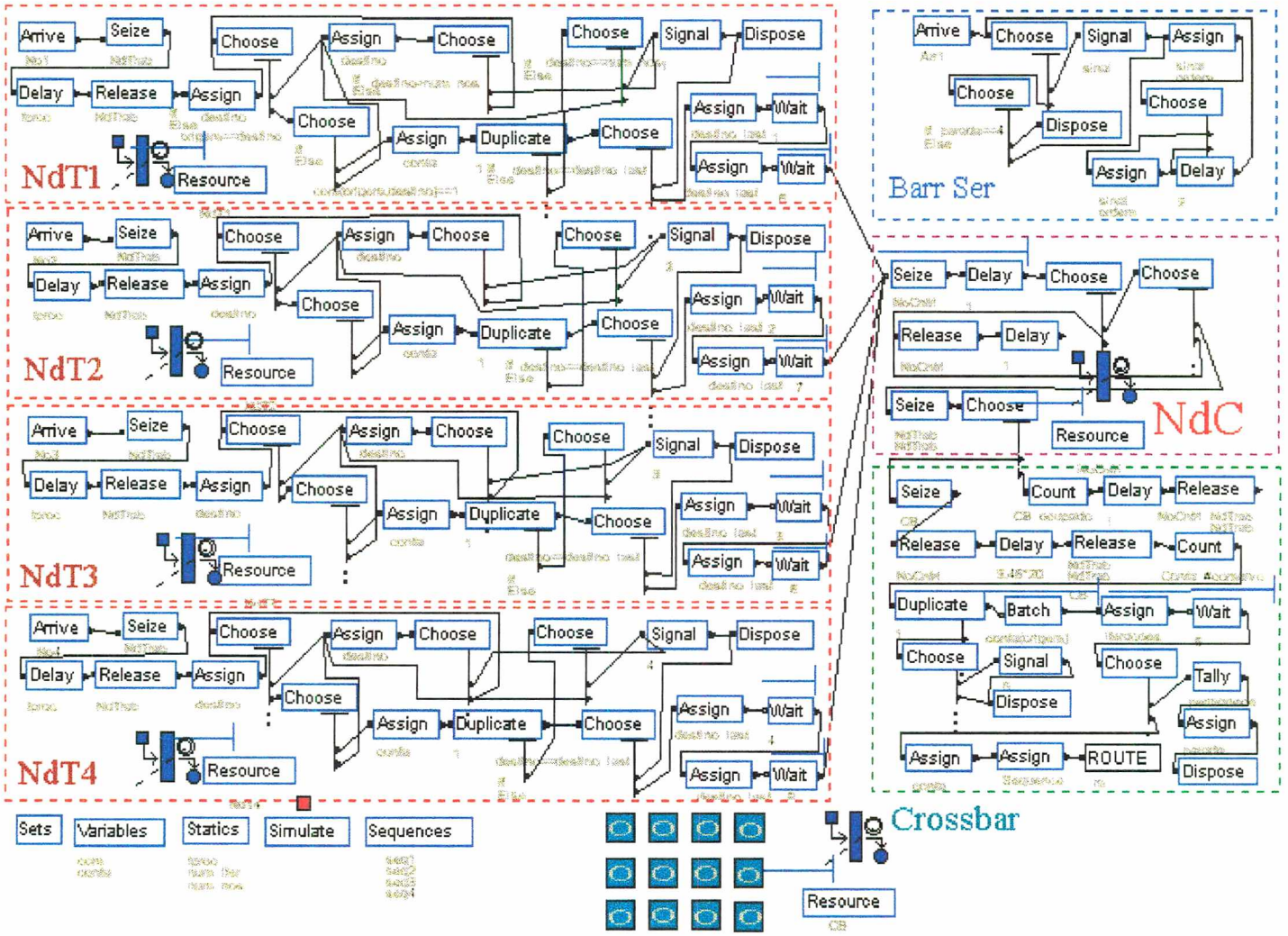


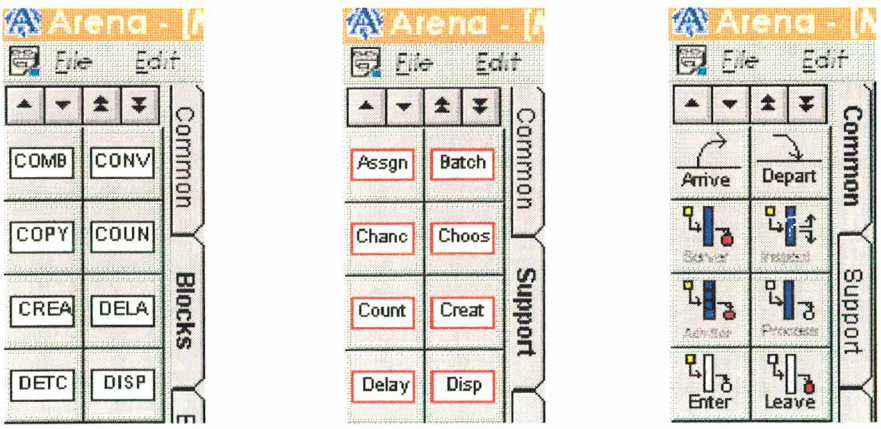
Figura 4-10 - Visão geral do modelo que simula o Nó// na arquitetura com Barramento de Serviço

Para os modelos apresentados, verificou-se que o número de módulos é excessivamente grande. Na modelagem para arquitetura com linhas de interrupção e com apenas quatro nós de trabalho, o modelo lógico apresentou 113 módulos. Para a arquitetura com barramento de serviço, 126 módulos básicos do ARENA. Mais uma vez é importante ressaltar que estes modelos possuem apenas quatro nós de trabalho, e no mercado existem máquinas com 128 processadores. Para exemplificar, um modelo de arquitetura com barramento de serviço e 128 processadores, se fosse construído desta maneira, possuiria o equivalente a 2.600 módulos. Isto vem comprovar a dificuldade que mesmo um projetista com conhecimentos do sistema encontra para modelar. A tarefa requer no mínimo que o usuário tenha dois tipos de especialização: um relativo ao próprio sistema e outro relativo à modelagem.

Capítulo 5

5. Templates

Um Template consiste de um painel ou um conjunto de painéis que abrange construções para modelagem de um sistema sob investigação. Um painel é um conjunto de módulos ou blocos que é utilizado para a construção do modelo. O Template da linguagem SIMAN é composto por dois painéis, denominados de *Blocks* (Figura 5-1a) e *Elements* e na linguagem ARENA, consiste de três painéis: *Support* (Figura 5-1b), *Commom* (Figura 5-1c), e *Transfer*.



(a) (b) (c)

Figura 5-1- Exemplos de Templates - SIMAN - Blocks (a), ARENA - Support (b) e Common (c)

A versão profissional do ARENA permite o desenvolvimento de novos Templates, os quais podem ser específicos para a modelagem de novas funções ou sistemas. Esta arquitetura aberta permite o uso da tecnologia oferecida pela linguagem e sua interface de programação.

Após o desenvolvimento de um novo Template, ele pode ser agregado a qualquer modelo que utilize os templates padrões que são fornecidos com o ARENA.

5.1 Metodologia para a Construção de um Template

A sistemática colocada, a seguir, apresenta os passos necessários para a concepção de um novo Template. No primeiro item coloca-se a descrição do sistema, o desenvolvimento inicial e a preparação dos dados que o Template deve englobar. A seguir a montagem de um modelo, que posteriormente será analisado e dividido. Cada divisão obedece certos procedimentos que compõem as etapas de construção, e após a compilação destes dados, conclui-se um novo módulo dentro do Template. A parte final é a montagem de um modelo utilizando os módulos criados para validar o Template desenvolvido.

Esta metodologia é dividida em cinco passos, apresentados a seguir:

- descrição minuciosa do sistema;
- montagem e análise do modelo;
- divisão do modelo;
- construção de um módulo;
- validação.

Estes passos são importantes para a concepção da ferramenta e serão detalhados nos itens seguintes.

5.1.1 Descrição do sistema

A descrição do sistema é um conjunto de tarefas que demanda tempo em pesquisa e deve ser obedecido em todos os aspectos, que são: elaboração de um relatório com as peculiaridades do sistema, ambiente de operação, fluxograma de operação, relatórios de tempo, figuras e tabelas. A obrigatoriedade destes itens depende do sistema a ser analisado.

Para a modelagem do Nó// em cada uma das configurações, foi feito o levantamento do funcionamento detalhado do sistema, que é apresentado a seguir. Para configurar o modelo, cada nó recebe uma identificação denominada de origem, o tempo de processamento inicial, quantas variáveis cada comunicação vai processar e o número de iterações que o sistema irá realizar. Um vetor configurado em cada um dos nós de trabalho, com dimensão igual ao número de nós, receberá informações dos destinos das comunicações. Neste caso, se o nó de trabalho 1 deseja efetuar comunicações com o nó de trabalho 2 e mais nenhum, seu vetor de comunicações terá a seguinte seqüência: 0,1,0,0. Uma lógica verifica se o programador não cometeu algum engano e colocou o destino igual a origem, o que não poderá acontecer.

A diferença lógica da modelagem entre a arquitetura com linhas de interrupção e barramento de serviço é descrita a seguir. O nó de controle pode receber a requisição de comunicação via linha de interrupção ou aguardar o *polling* via barramento de serviço e verificar quem deseja se comunicar. Estando notificado de uma maneira ou de outra, o nó de controle processa a mensagem carregada destes atributos iniciais e verifica se o nó destino não está ocupado, processando ou comunicando, e se há vagas para o estabelecimento de comunicações no comutador de conexões ou *crossbar*. Este último pode ser configurado quanto à sua capacidade nos seguintes valores: 4, 6, 8, 12, 16 ou 32. Caso ocorra a impossibilidade de comunicação, a mensagem aguarda numa fila. O tempo de simulação do processamento das variáveis será acrescentado do tempo de espera nesta fila até a liberação do recurso (*crossbar* ou nó de trabalho), para efetuar a comunicação.

5.1.2 Montagem do modelo

A montagem do modelo foi apresentada com detalhes no capítulo 4, e foi baseada nas informações obtidas na pesquisa do funcionamento do sistema. Nesta montagem realizada, os conceitos básicos de modelagem são aplicados, tomando-se a precaução de simplificar o mesmo para capturar informações necessárias e descartar informações que não venham acrescentar dados importantes.

5.1.3 Divisão do modelo

Procura-se nesta etapa do processo de criar um novo módulo, elementos que exerçam funções semelhantes ou que possuam as mesmas características. Estes elementos são desagregados e farão parte de um sub-modelo. Cada sub-modelo é submetido a um nível de agregação maior, até chegar a um módulo representativo que exerça a mesma função do originador. Por trás deste novo módulo existirá uma lógica semelhante à que existia no sub-modelo.

5.1.4 Construção do módulo

O desenvolvimento de um módulo no ambiente do ARENA *Professional Edition* obedece etapas bem caracterizadas. Cada uma destas etapas é desenvolvida dentro de uma circunstância bem característica, às quais serão nomeadas de janelas. Cada uma destas janelas conduzem à construção de todos os elementos pertinentes a um módulo e são definidas como segue:

- janela de definição dos operandos ou janela dos operandos;
- janela de definição da lógica;
- janela de chaveamento;
- janela visão do usuário;
- editor do ícone do painel.

Na janela dos Operandos define-se a interface do módulo: tamanho das lacunas onde se inserem os dados, seqüência de entrada, dados permitidos e tipos de conexões com outros módulos. A construção é feita de forma gráfica, e permite a entrada de um dado simples, dados múltiplos, bem como a abertura de novas janelas em um segundo nível. O resultado final desta janela é o que será mostrado ao usuário. Este tem acesso à janela dos Operandos apenas para fornecer os dados, variáveis e atributos utilizados pelas entidades e que pertencem à lógica do novo módulo.

Na janela Lógica, como o próprio nome diz, tem-se a lógica utilizada no módulo a ser criado, é um mini-modelo que não será acessível ao usuário final. Esta lógica possui uma estreita relação com os dados fornecidos na primeira janela (Operandos).

Na janela seguinte, Visão do Usuário, será a concepção do que aparece na tela no momento em que o módulo é agregado ao modelo. É nesta janela que se define a animação que será adicionada ao modelo, se houver.

A janela de chaveamento é uma janela de construção auxiliar que permite controlar a informação que aparece ao projetista onde apenas os campos relevantes são mostrados. Pode ser um artifício para controlar uma seqüência de entrada de dados. A construção nesta janela só é necessária quando, dentro da caixa de diálogo, houver uma escolha a ser feita.

A última janela é destinada à construção do ícone que aparece na tela no momento em que o painel é agregado à tela de trabalho. A figura a ser construída pode ser importada de arquivos *.dxf* ou desenhada com ferramentas disponíveis na tela de trabalho.

Ao final deste trabalho realizado nas janelas de construção, verifica-se a ocorrência de algum erro através de uma ferramenta disponível no programa e compila-se o novo módulo.

5.1.5 Validação

Ao final da montagem do Template, é necessário validá-lo com testes que garantam a sua funcionalidade. Apresenta-se a seguir uma maneira de realizar este teste.

O projetista pode efetuar um teste construindo um modelo que utiliza os módulos criados no novo Template. Após o término da montagem, são realizadas replicações e a coleta dos resultados. Estes resultados podem ser comparados com relatórios de um modelo anterior ou dados extraídos de outra simulação.

5.2 Aplicação da Metodologia

Neste item será apresentado um exemplo de aplicação da metodologia para elaboração de um Template. As primeiras etapas da metodologia elaborada, que seriam definição das diretrizes de funcionamento, fluxograma de funcionamento e construção do modelo propriamente dito, já foram efetuadas no capítulo 4. O modelo construído foi apresentado na Figura 4-8 para linhas de interrupção e Figura 4-10 para barramento de serviço.

A etapa de agrupamento de módulos que executam a mesma função foi realizada com os módulos que possuem a tarefa de geração inicial das entidades e a este grupo denominou-se nó de trabalho. Ao todo são quatro repetições de blocos que posteriormente serão substituídas por quatro módulos, cada um nomeado nó de trabalho com uma identificação para diferenciá-los (Figura 5-2).

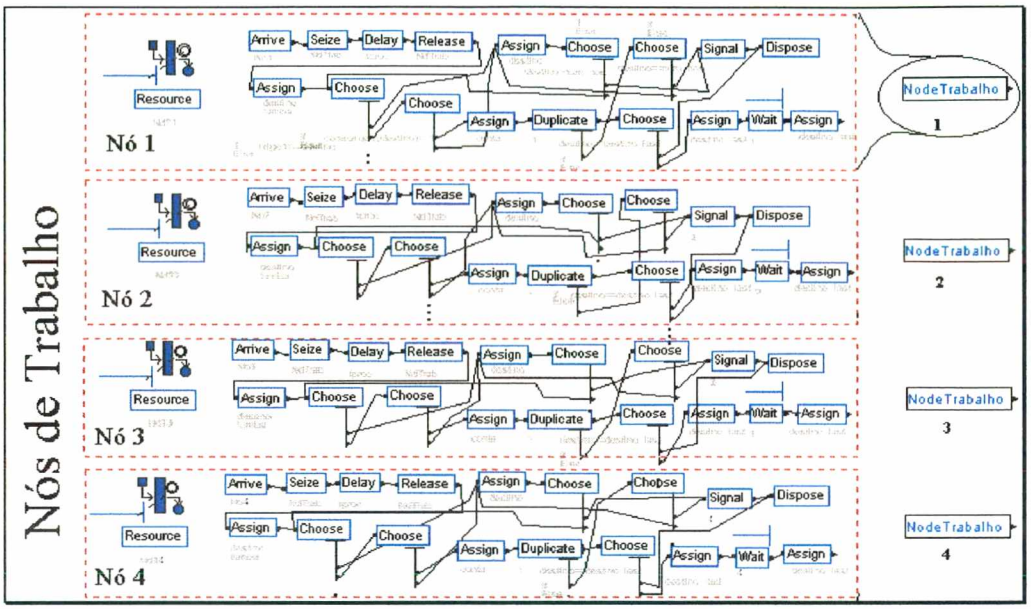


Figura 5-2 - Divisão dos módulos ressaltando o nó de trabalho 1

A Figura 5-2 apresenta a divisão dos módulos que realizam funções semelhantes. Com a agregação destes 19 elementos formando um sub-modelo e passando pelas etapas de confecção de um novo módulo, eles resultarão em um modelo simplificado de apenas 4 módulos, como pode-se observar ao lado direito da figura.

As cinco etapas que compõem a metodologia são realizadas buscando a concepção de um módulo que fará parte do Template. A seguir são detalhados os procedimentos executados em cada etapa com as respectivas ilustrações.

A janela inicial na construção da ferramenta é a definição de todos os módulos que farão parte deste Template. Para realizar as funções deste exemplo, temos cinco módulos

determinados: definição dos NTs, *crossbar*, nó de trabalho, comunicações e nó de controle. A Figura 5-3 ilustra a janela inicial para inclusão dos módulos componentes.

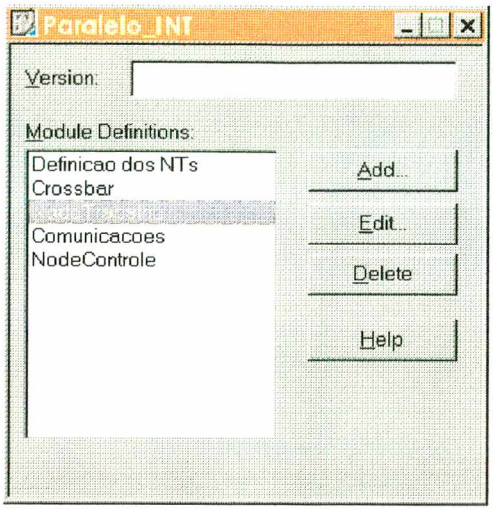


Figura 5-3 - Definição dos Módulos que farão parte do Template

Após a definição de todos os módulos, cada um em particular deverá ter sua apresentação, sua lógica, seu ícone e assim por diante. Estas características são colocadas na seqüência de janelas "operandos", "lógica", "chaveamento" e "ícone". Para definir a entrada de dados neste módulo "nó de trabalho", na janela dos "operandos" define-se a interface do módulo: tamanho das lacunas onde se inserem os dados, seqüência de entrada, dados permitidos e tipos de conexões com outros módulos (Figura 5-4).

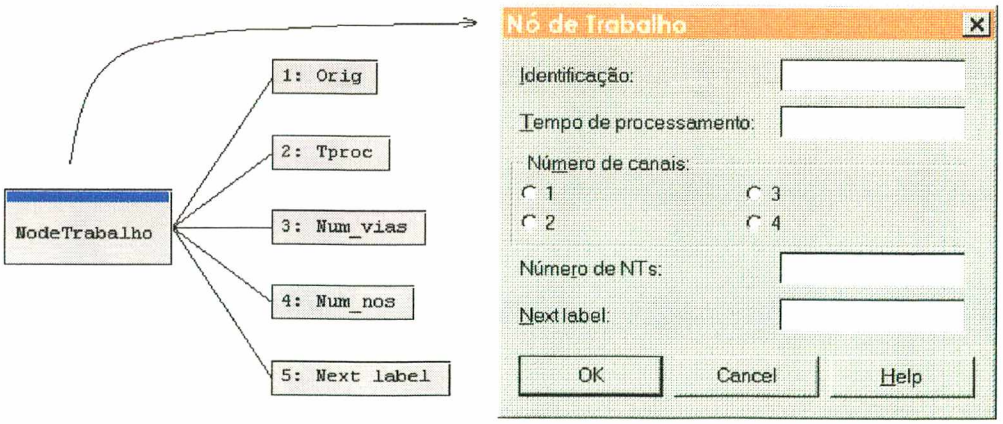


Figura 5-4 - Janela Operandos e "*Preview*"

A construção é feita de forma gráfica, e permite a entrada de um dado simples, como pode-se observar na Figura 5-4 no item "identificação". Pode-se ainda entrar com dados pré-

estabelecidos como no item "número de canais", onde seleciona-se um número entre 1 e 4. O usuário tem acesso apenas ao produto final, não podendo alterar a forma de entrada dos dados.

Na janela "lógica", como o próprio nome diz, temos a lógica utilizada no módulo a ser criado, é um mini-modelo que não será acessível ao usuário final (Figura 5-5). Esta lógica possui uma estreita relação com os dados fornecidos na primeira janela (Operandos).

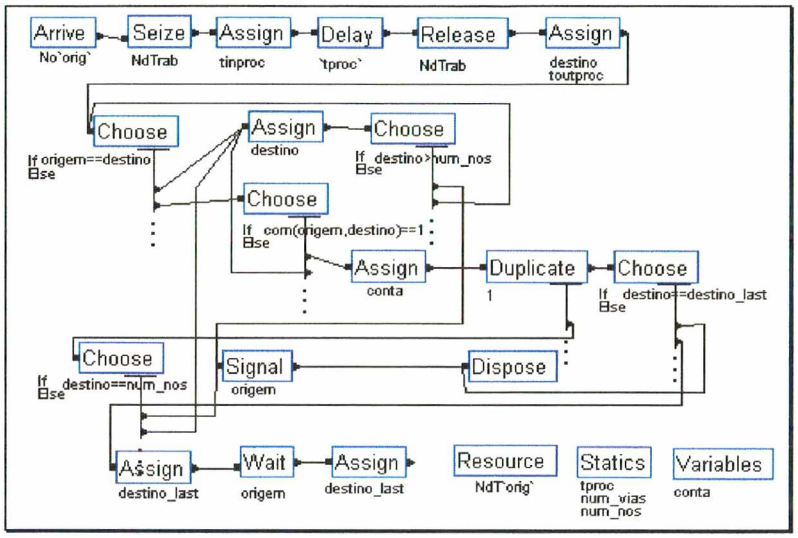


Figura 5-5 - Janela Lógica

Na janela seguinte, denominada de "visão do usuário", tem-se a concepção do que aparece na área de trabalho no momento em que o usuário agrega o módulo ao modelo. É nesta janela que se define a animação que será adicionada ao modelo, se houver. O que vai aparecer na tela de trabalho do usuário no momento em que ele agregar o módulo nó de trabalho ao modelo, está mostrado na Figura 5-6.

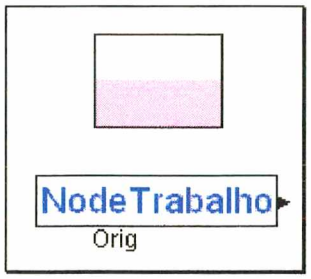


Figura 5-6 - Janela Visão do Usuário

Na parte inferior apresenta-se a sigla Orig, que vai ser substituída pela Identificação colocada pelo usuário. Por exemplo, se o usuário chamá-lo de nó de trabalho "1", então,

abaixo do módulo, aparecerá o número "1". O quadrado em rosa significa a animação correspondente ao recurso "nó de trabalho".

A janela de chaveamento é a etapa da construção do template que permite controlar a informação que aparece ao projetista, dependendo da seleção efetuada. É uma maneira elegante de evitar uma seqüência de caixas de diálogos desnecessária (Figura 5-7).

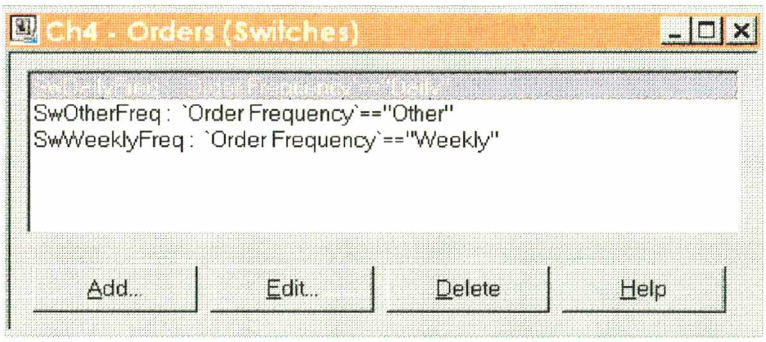


Figura 5-7- Janela de Chaveamento

Para exemplificar, descreve-se um chaveamento que utiliza um sistema com taxa de chegada de pacotes. Esta taxa pode ser de três maneiras diferentes: diariamente (Daily), semanalmente (Weekly) ou outra opção (Other) Figura 5-8).

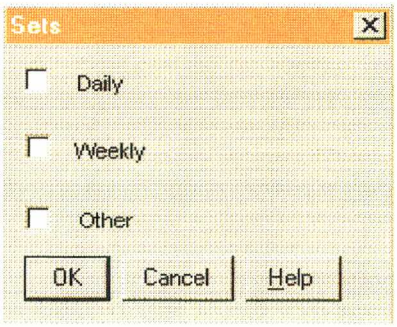


Figura 5-8- Exemplo de *dialog box* onde ocorre chaveamento

Quando o usuário escolher uma das três opções, por exemplo "daily", aparecerá uma nova janela que receberá a taxa de chegada de maneira diária. Se a opção for "weekly" (semanalmente), outra janela surgirá para entrar com a taxa de chegada por semana, sendo que a última opção ("other") seria uma outra taxa, como por exemplo, a mensal.

A última janela é destinada à construção do ícone que aparece na tela no momento em que o painel é agregado à tela de trabalho (Figura 5-9). A figura a ser construída pode ser importada de arquivos *.dxf* ou desenhada com ferramentas disponíveis na tela de trabalho.

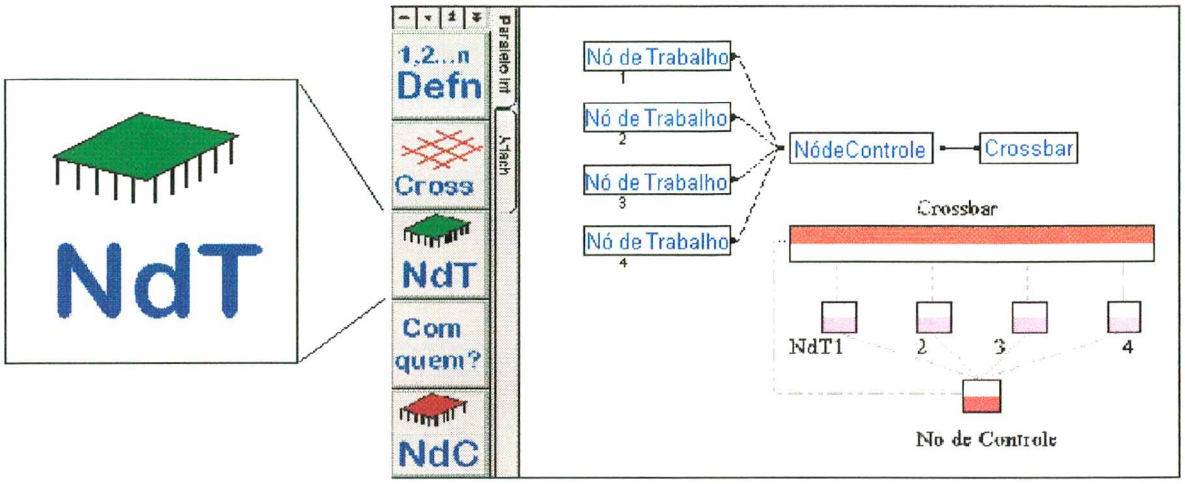


Figura 5-9 - Janela Ícone e sua posição na área de trabalho do ARENA ("work area")

Ao final desta seqüência de janelas auxiliares de construção, verifica-se a ocorrência de algum erro através da ferramenta *check* e compila-se o novo módulo.

Utilizando a metodologia descrita neste capítulo, foi desenvolvida a ferramenta que é objeto deste trabalho. Ela será apresentada no próximo capítulo, em telas ilustrativas mostrando como é realizada a inserção de dados no modelo.

Capítulo 6

6. Ferramenta desenvolvida

Neste capítulo será apresentada a ferramenta desenvolvida que permite simular o multicomputador Nó// em suas duas arquiteturas propostas: com linhas de interrupção e com barramento de serviço. Para o desenvolvimento desta ferramenta foi elaborada uma metodologia descrita no capítulo anterior dentro do ambiente de modelagem fornecido pelo software ARENA 2.1. Serão apresentadas as interfaces de comunicação com o usuário, denominadas *dialog box,* e as instruções para inserção dos dados e montagem do modelo. Ao final será apresentada a validação das ferramentas: Paralelo INT e Paralelo BUS.

6.1 Apresentação da Ferramenta Paralelo INT

A ferramenta Paralelo INT faz parte do Template Paralelo e foi desenvolvida visando a modelagem e simulação de um sistema paralelo que utiliza linhas de interrupção e ao mesmo tempo voltada para a redução do tempo entre a concepção do projeto e a sua análise. Neste item serão mostrados os ícones, a construção do modelo e as caixas de diálogo com as instruções para a introdução de dados no template Paralelo INT.

6.1.1 Ícones

Os ícones representam os módulos no menu lateral da interface ARENA. Ao selecionar Template Paralelo INT, visualizam-se os ícones Definição dos nós de trabalho, *crossbar* ("grade púrpura"), nó de trabalho ("*chip* verde"), nó de controle ("*chip* vermelho") e comunicações ("com quem?"), veja no lado esquerdo da Figura 6-1. É a partir desta visualização de figuras pertencentes ao ambiente que o usuário está acostumado a usar no seu dia a dia, que o uso dos Templates torna fácil a tarefa de montagem de um modelo.

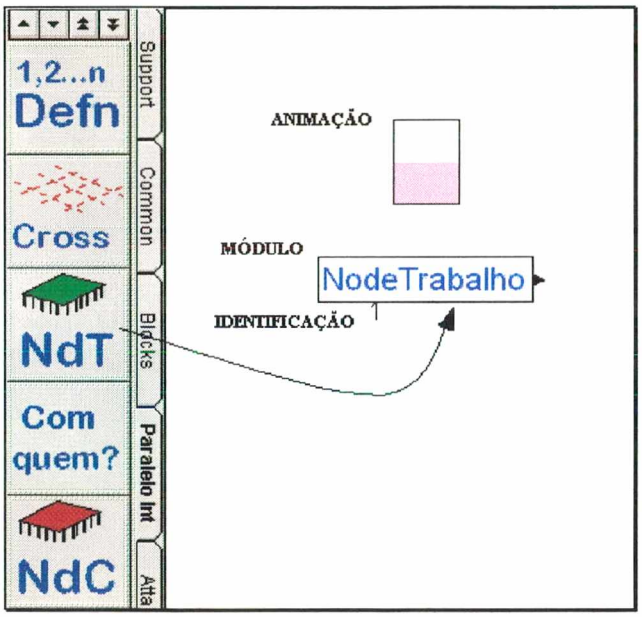


Figura 6-1 - Ícones do Template Paralelo INT e a área de trabalho

Seleciona-se o módulo desejado e dirige-se para a área de trabalho para compor o modelo em construção. Se o módulo possuir uma figura de animação, esta desloça-se juntamente para a área de trabalho. Durante a simulação do modelo a animação vai ser preenchida com a coloração, se o recurso representado pelo módulo estiver realizando alguma operação.

6.1.2 Construção do modelo

Será construído um modelo com oito nós de trabalho para demonstrar como é simples utilizar a ferramenta. Para continuar a construção do modelo, outros módulos são agregados à área de trabalho. No exemplo que está sendo mostrado, serão agregados mais sete "nós de trabalho", que serão conectados ao "nó de controle". Este, por sua vez, será conectado ao "*crossbar*". Os outros dados de entrada do modelo serão colocados nos módulos "definição dos nts" e "comunicações".

É o que mostra a Figura 6-2, um modelo com 8 nós de trabalho, nó de controle e *crossbar*.

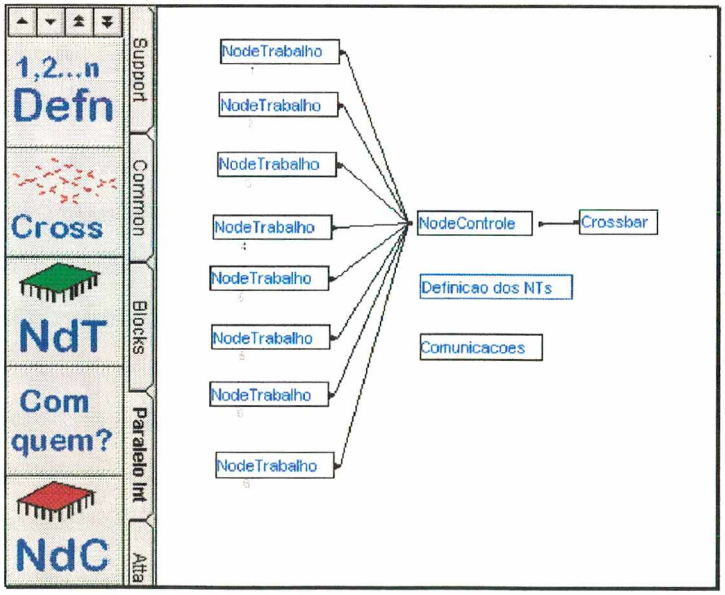


Figura 6-2 - Modelo de simulação de arquitetura paralela com 8 nós de trabalho

Os dados de entrada para cada módulo são colocados através da caixa de diálogo ou dialog box. Estes dados inseridos serão utilizados no momento da execução da simulação. Cada caixa de diálogo será mostrada em detalhes nos itens seguintes.

6.1.3 Caixas de diálogo

Por intermédio das caixas de diálogo, o usuário coloca os dados que completam o modelo, para realizar a simulação. A caixa de diálogo apresentada, a seguir, representa um nó de trabalho. Ele é a origem do modelo uma vez que as mensagens são criadas a partir dos dados nele inseridos. Estes dados fornecem as características referentes a cada nó de trabalho.

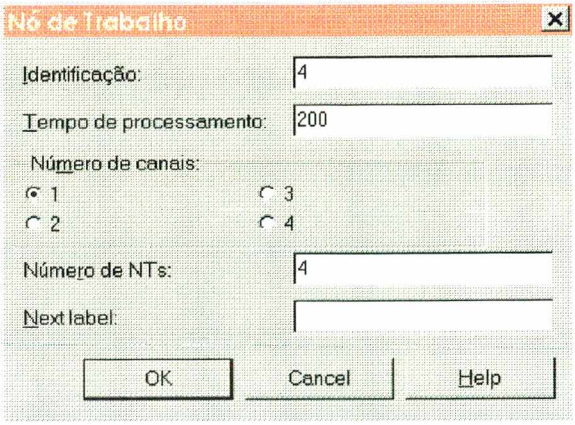


Figura 6-3 - Caixa de diálogo referente ao nó de trabalho

Observa-se na Figura 6-3 que o usuário pode escolher a identificação para cada nó. Nesta figura a identificação é representada pelo número 4. O segundo item é o tempo de processamento, que pode ser determinístico ou probabilístico. No primeiro caso seria, por exemplo, um número (100) e no segundo caso, uma distribuição exponencial (expo(300)). Neste exemplo, o tempo de processamento é determinístico e igual a 200. O número de canais que fazem a ligação do nó de trabalho ao crossbar pode ser selecionado entre 1 e 4. No último item preenche-se o número de nós de trabalho que o sistema irá possuir. Neste exemplo os tempos foram considerados em μs, unidade utilizada em trabalhos anteriores relativos ao projeto Nó//.

A caixa de diálogo descrita a seguir será do nó de controle. O dado único de entrada nesta caixa será o tempo de controle sobre as mensagens que circulam no sistema. Dentro desta caixa encontra-se também um box de ajuda, esclarecendo ao usuário as características

do dado a ser inserido. O tempo de controle obedece a unidade que o usuário está utilizando, e no exemplo da Figura 6-4 este tempo é de 8 μs.

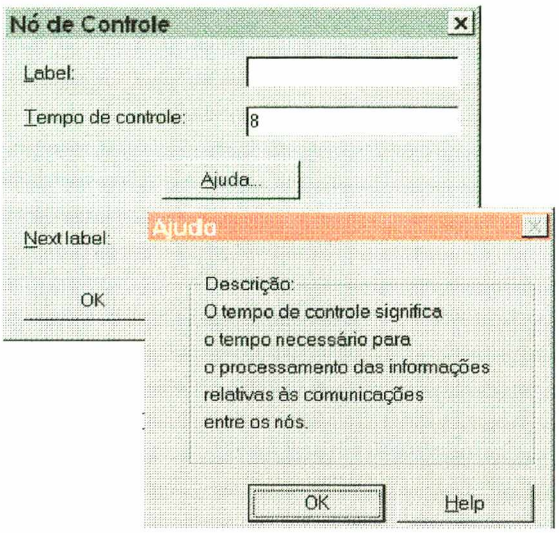


Figura 6-4 : Caixa de diálogo para o módulo nó de controle

Este tempo pode ser alterado de acordo com os dados da máquina a ser modelada e a unidade utilizada. Especificamente para a máquina do projeto Nó//, um tempo de controle de 88.7 μs foi determinado a partir de dados fornecidos pelo fabricante do Hardware [SIL96].

A caixa de diálogo que complementa a construção do modelo é a que trata das informações referentes ao *crossbar* (Figura 6-5). Nesta janela são colocados dados que fixam a capacidade do *crossbar*, colocam o número de variáveis que devem ser processadas em cada nó de trabalho e o número de iterações que o modelo deve realizar.

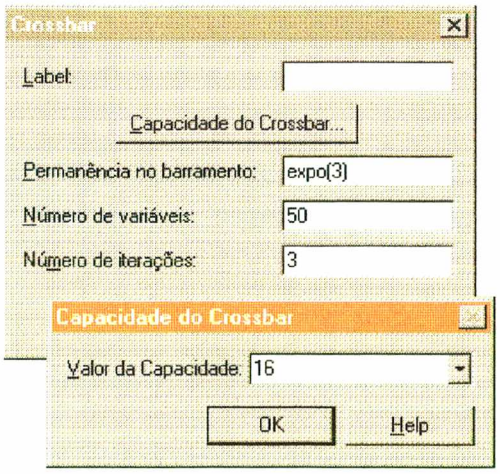


Figura 6-5 - Caixa de diálogo para o módulo *crossbar*

O *crossbar* recebe as mensagens e disponibiliza o meio físico para efetuar a comunicação. As entradas de dados que são apresentadas no exemplo foram preenchidas com os seguintes valores: a capacidade do *crossbar* selecionada a partir de valores usuais e, no caso, foi selecionado 16. O tempo de permanência no barramento é a unidade de tempo que o mesmo consome ao processar um byte. No exemplo este tempo é probabilístico segundo uma exponencial de média igual a três (expo 3). O número de variáveis que os nós de trabalho deverão processar está fixado em 50 e o número de iterações que o sistema deve realizar até divulgar o resultado final do processamento, em 3.

Além das caixas de diálogo correspondentes aos nós de trabalho, ao nó de controle e ao crossbar, faz-se necessário, por parte do usuário, a identificação de cada nó de trabalho. Esta declaração de identidade será comum a todo o sistema.

É mostrado na Figura 6-6, a entrada dos dados na caixa de diálogo principal da definição dos nós de trabalho. A identificação poderá ser tanto numérica quanto alfa-numérica. Na figura são seis nós de trabalho, numerais seqüenciais de 1 a 6.

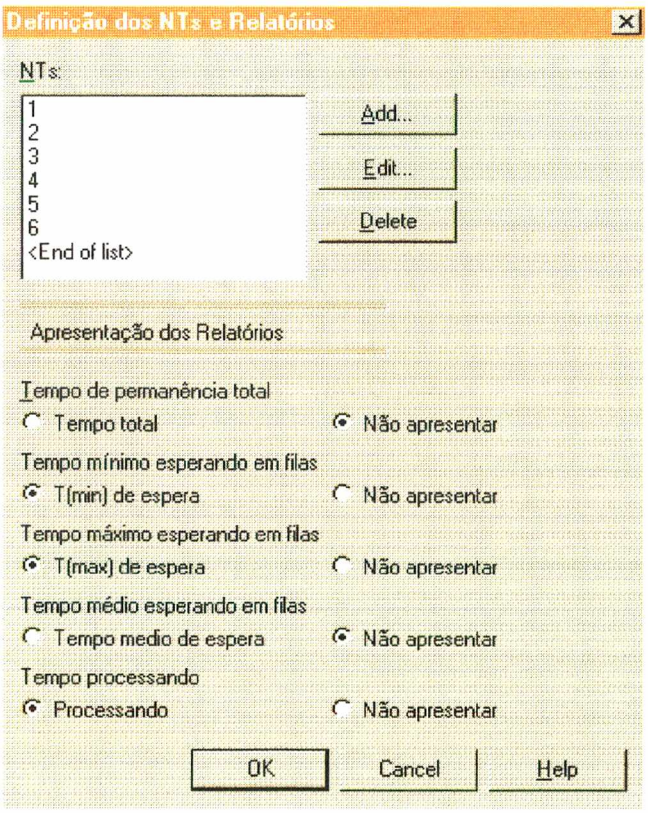


Figura 6-6 - Definição dos Nós de Trabalho e apresentação dos Resultados

Para emissão de relatórios o ARENA possui uma maneira particular de colocar os dados para o usuário. Estes dados aparecem em colunas e de uma forma parametrizada. Foi criada uma interface mais informal para apresentação destes resultados e é o próprio usuário quem seleciona os itens que constarão no relatório final.

Para determinar o destino das comunicações foi construída uma matriz de vetores com linhas denominadas com a sigla "Ori" de Origem mais um número indicativo da linha e colunas com a sigla "Des" da palavra destino mais um número indicativo da coluna correspondente.

O preenchimento desta caixa de diálogo deve proceder da seguinte maneira: inicialmente será dado um nome a esta matriz (Figura 6-7); os elementos que a compõem só podem ser "0" e "1". Para preenchê-la deve ser seguida a seguinte regra: a intersecção do nó origem que deseja efetuar comunicação com o nó destino receberá o número "1". O contrário, preenchimento com "0", representa que o nó origem não efetua comunicação com o destino. Por exemplo: se o Nó1 deseja comunicar-se com o Nó2, na linha da matriz correspondente ao Nó1 a seqüência de preenchimento é a seguinte: 0,1,0,0,0,0,0,0.

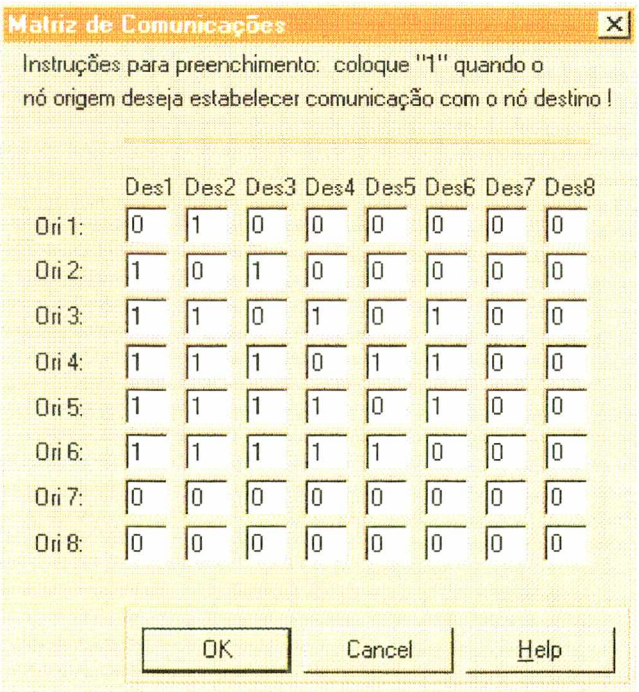


Figura 6-7- Matriz de vetores de comunicação

Após apresentar e preencher as caixas de diálogo que compõem o modelo em questão, mostra-se como se pode aproveitar para colocar a animação no momento da simulação. A animação é uma caixa colorida agregada ao modelo junto com o módulo correspondente, representando seu recurso (Figura 6-8).

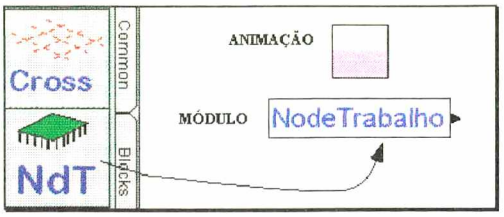


Figura 6-8 - Animação agregada ao módulo

Este modelo onde aparecem as animações possui seis nós de trabalho e apenas um canal de comunicação entre cada nó e o *crossbar* (Figura 6-9).

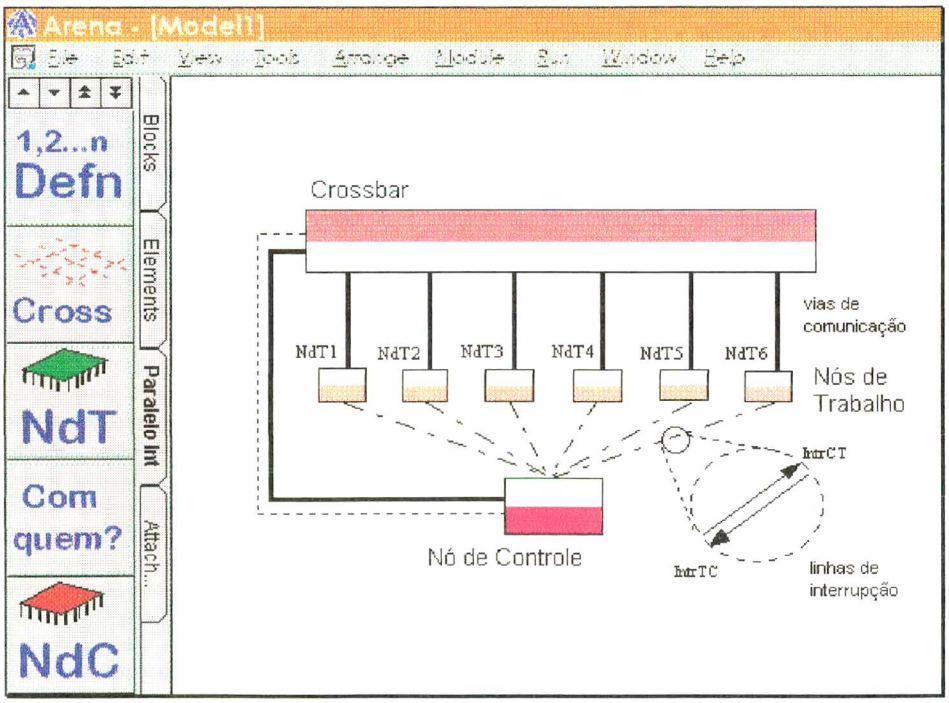


Figura 6-9 - Modelo com animação de arquitetura com linhas de interrupção

Além da animação foi colocado um detalhe chamando atenção das linhas de interrupção, que são nos dois sentidos, INTtc - nó de trabalho para nó de controle e INTct - nó de controle para nó de trabalho.

6.2 Apresentação da Ferramenta Paralelo BUS

A ferramenta Paralelo BUS faz parte do Template Paralelo e foi desenvolvida para reduzir o tempo entre a concepção do projeto e a modelagem e simulação de um sistema paralelo que utiliza barramento de serviço. Visto que os ícones e as caixas de diálogo são semelhantes à ferramenta anteriormente apresentada, neste item será mostrada apenas a janela do barramento de serviço que é uma característica particular desta arquitetura. Os procedimentos descritos para agregar o painel e preencher os campos nas caixas de diálogo podem ser vistos no item anterior 6.1.

Obedecendo a um mesmo padrão do Template Paralelo INT, o Template Paralelo BUS apresenta a mesma interface, porém acrescenta mais um módulo em seu painel.

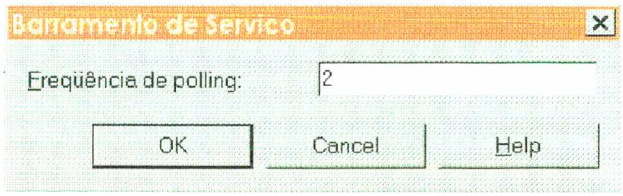


Figura 6-10 - Módulo para barramento de serviço

A freqüência de *polling* é a unidade de tempo decorrida entre cada varredura que o nó de controle vai efetuar nas portas de cada nó de trabalho para verificar a presença ou não de mensagens (Figura 6-10). Estas mensagens possuem a informação do destino da comunicação que é retirada da matriz de vetores vista anteriormente.

Como foi apresentado para a arquitetura com linhas de interrupção, para barramento de serviço o modelo também pode possuir animação. Neste exemplo foram colocados 6 nós

de trabalho com 4 vias de comunicação com o crossbar (Figura 6-11).

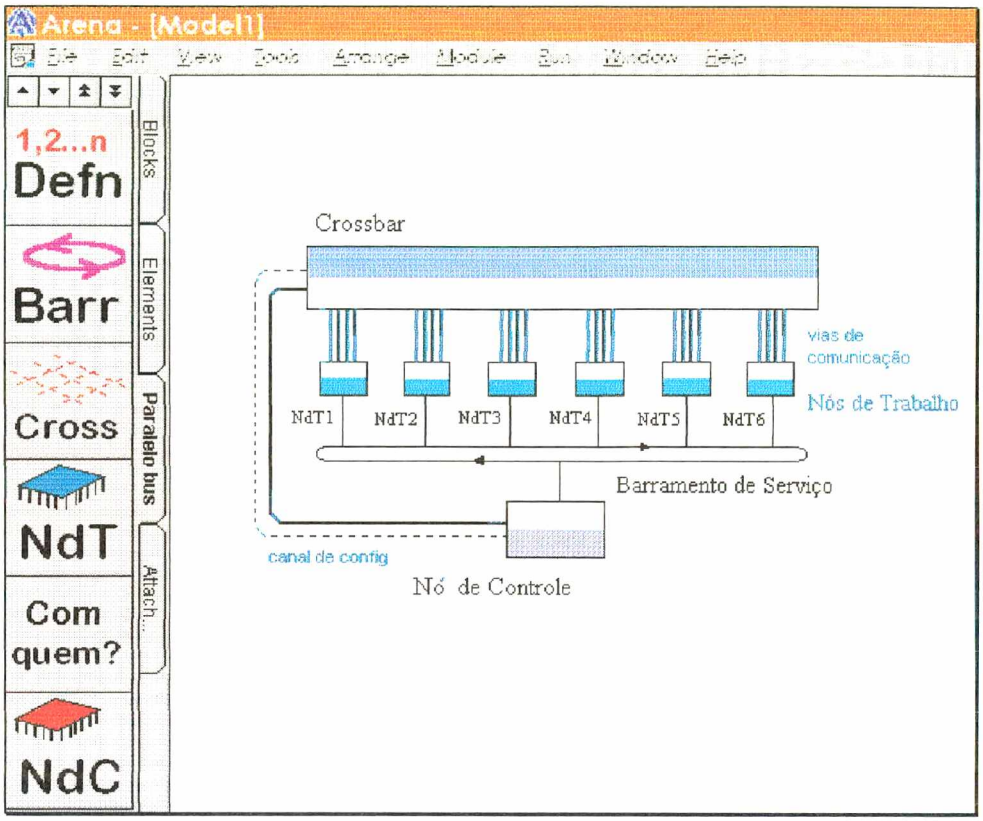


Figura 6-11 - Ilustração de animação em arquitetura com barramento de serviço

Terminando a apresentação dos dois painéis desenvolvidos, segue-se agora à validação da ferramenta onde são apresentadas as variáveis voltadas à analise de desempenho do modelo.

6.3 Validação da ferramenta

As variáveis voltadas à análise de desempenho em computadores que apresentam arquitetura paralela são várias. Entre estas, encontram-se as seguintes:

- taxas de ocupação dos nós de trabalho, nó de controle e *crossbar*;
- tempo médio de espera das mensagens no nó de controle;
- número de mensagens que não conseguem tomar o *crossbar*;

- tempo de processamento e tempo de permanência no sistema.

Para a validação da ferramenta, foram coletados em determinados experimentos algumas destas variáveis e comparadas com trabalhos anteriores desenvolvidos sobre o tema [FRE96] [MER96] [SIL96]. Os resultados desta validação encontram-se no Anexo.

Capítulo 7

7. Conclusões

Em pesquisas e trabalhos com sistemas computacionais, no que se refere à realização da avaliação de desempenho, necessita-se do apoio de ferramentas de análise. Este trabalho apresentou o desenvolvimento de uma ferramenta destinada à modelagem, simulação, experimentação e análise de sistemas com arquitetura paralela. A análise destes sistemas envolve cálculos analíticos complexos em teorias de filas que necessitam de um auxílio computacional, seja por intermédio de ferramentas, linguagens específicas ou softwares matemáticos.

Este trabalho envolveu as seguintes áreas: processamento paralelo, simulação e modelagem, apresentando taxonomias dentro de máquinas computadoras, técnicas existentes para a avaliação de sistemas e linguagens de simulação. Foram colocadas as vantagens e desvantagens de cada linguagem e foi dado ênfase à linguagem de simulação utilizada: ambiente de simulação do ARENA.

Foi apresentado o Projeto Nó //, nome dado à máquina com arquitetura paralela em desenvolvimento no Curso de Pós Graduação em Ciência da Computação da Universidade Federal de Santa Catarina, descrição geral da máquina, citando-se as duas configurações básicas, e mostrados os componentes do multicomputador e o seu funcionamento nas arquiteturas com barramento de serviço e linhas de interrupção.

Um ambiente para modelagem de propósito geral foi caracterizado, e ressaltada a dificuldade que um projetista, mesmo com conhecimentos do sistema, encontra para modelar. Há a necessidade de possuir, no mínimo, dois tipos de especialização: um relativo ao próprio sistema e outro relativo à modelagem. E este foi um dos motivos levados ao desenvolvimento desta ferramenta, para auxiliar o projetista e diminuir o tempo existente entre o projeto e o modelo para simulação.

A ferramenta foi confeccionada segundo uma metodologia desenvolvida e relatada neste trabalho em detalhes, envolvendo as etapas para construção, dentro do ambiente de simulação ARENA versão 2.1. Esta foi uma parte que, no desenvolvimento do trabalho, dispendeu muito tempo implementando-se a modelagem das duas arquiteturas, realizando-se testes para a validação dos modelos e dividindo-se o modelo geral em sub-modelos. A dificuldade encontrada nesta última parte citada da elaboração do trabalho foi concatenar a lógica e as diversas variáveis dentro de cada sub-modelo utilizado para a elaboração final do Template. A parte da interface de comunicação com o usuário referente a cada módulo (*dialog box*) também envolveu certos cuidados para não sobrecarregar a janela com informações, e o produto final somente foi elaborado depois de executados testes de apresentação. Estes testes são relativos à compilação da ferramenta e elaboração de um pequeno modelo para a verificação da interface gráfica.

Após a elaboração da ferramenta, foram realizados testes de validação e os resultados então comparados com o modelo geral desenvolvido inicialmente e modelos construídos em trabalhos anteriores relacionados com aplicações nas duas arquiteturas que foram propostas: com linhas de interrupção e com barramento de serviço.

As contribuições deste trabalho situam-se nas áreas de avaliação de desempenho, processamento paralelo e algoritmos paralelos. O Template pode ser utilizado na elaboração de modelos para simulações de programas paralelos como redes de processos comunicantes, como por exemplo, serviços gerais de um sistema de computação:

- sistemas operacionais - simulação de um modelo que represente a organização de um sistema operacional como uma implementação em uma rede de processos comunicantes utilizando a máquina paralela;

- servidores de arquivos - simulação de um modelo que represente um sistema de arquivos não monolítico concebido como um servidor e implementado como uma rede de processos comunicantes com grande autonomia em relação ao resto do sistema;

- compiladores - simulação de um modelo que represente um compilador executando paralelamente as fases distintas de análise léxica, análise semântica, análise sintática e geração de código.

Visto que a limitação desta ferramenta está no fato de se aplicar em arquiteturas que apresentem configuração com linhas de interrupção ou com barramento de serviço, a perspectiva para novas pesquisas estende-se a outras arquiteturas que se aplicam ao paralelismo.

Outra limitação que ocorre normalmente em trabalhos que envolvem tecnologia de ponta é a evolução constante das máquinas, com novas características e implementações, e que podem não estar contempladas nas opções oferecidas pela ferramenta desenvolvida. Isto resulta em uma nova frente de pesquisa, buscando atualização constante e geração de novos Templates para contemplar estas inovações.

8. Bibliografia

[BAK97] BAKER, G. S., *Taking the work Out of Simulation Modeling: An Application of Technology Integration.* Winter Simulation Conference (WSC97)-1997.

[BEL94] BELL, G. *Scalable, Parallel Computers: Alternative, Issues and Challenges.* Intl. Journal of Parallel Programming. v.22 nº 1, p 3-47 - 1994.

[CAM95] CAMPOS, R. A. *Um sistema operacional fundamentado no modelo cliente-servidor e um simulador multiprogramado para multicomputador.* Dissertação de Mestrado, Curso de Pós-Graduação em Ciências da Computação, Universidade Federal de Santa Catarina, Florianópolis, junho de 1995.

[COR93] CORSO, T.B. *Ambiente para programação paralela em multicomputador. Relatório Técnico nº1*, Departamento de Informática e de Estatística, Universidade Federal de Santa Catarina, Florianópolis, novembro de 1993.

[DUN90] DUNCAN, R. *A survey of parallel computer architectures.* IEEE Computer. V.23, nº 2, p.5-16. Feb 1990.

[FIS95] FISHWICK, P.A. *Simulation Model Design and Execution, Building Digital Words* - New Jersey Prentice Hall - 1995.

[FRE96] FREITAS Fº, P.J., MERKLE, C., ZEFERINO, C.A., BOING, H., SILVA, V.A. *A Simulation Model for the Comparison of Two Multicomputer Architectures*, Portland - Oregon SCSC - 1996.

[FLY72] FLYNN, M. J., *Some Computer Organizations and Their Effectiveness.* IEEE Trans. On Computers, vol. C-21, p. 948-960, Setembro de 1972.

[HAN95] HANSEN, P. B. *Studies in Computational Science - Parallel Programming Paradigms* Prentice Hall, New Jersey - 1995.

[HEI74] HEIDORN, G.E. "*English as a very high level language for simulation programming*" SIGPLAN Notices, vol 9, no 4, pp 91-100 -1974.

[HWA93] HWANG, Kai. *Advanced computer architecture : parallelism, scalability, programmability*. McGraw-Hill, Singapore - 1993.

[JAI91] JAIN, R. "*The Art of Computer Systems Performance Analysis*" Jonh Wiley & Sons, Inc - USA 1991.

[KLE77] KLEINE, H., "*A vehicle for developng standards for simulation programming.*" in Proccedings of the 1977 Winter Simulation Conference, Gaithersburg, MD, pp. 730-741 - 1977.

[LAW91] LAW, A. M. and KELTON, W. D. Simulation modeling and analysis 2nd ed. Mc Graw-Hill EUA - 1991.

[MAT74] MATHEWSON, S. C., "Simulation Program Generators", Simulation, vol 23, no.6 pp. 181-189 - 1974.

[MER96] MERKLE, C & BOING, H *Simulação do Nó //. Relatório Técnico*, Departamento de Informática e de Estatística, Universidade Federal de Santa Catarina, 1996.

[MID96] MIDORIKAWA, E.T., SATO, L.M., SENGER, *Introdução à Programação Paralela e Distribuída*, XV Jornada Atualização em Informática, Recife PE, 1996.

[MON95] MONTEZ, C. B. *Um sistema operacional com micronúcleo distribuído e um simulador multiprogramado de multicomputador*. Dissertação de Mestrado, Curso de Pós-Graduação em Ciências da Computação, Universidade Federal de Santa Catarina, Florianópolis - 1995.

[NAN77] NANCE, R. E., "*The feasability of and methodology for developing federal documentation standards for simulation models.*" Final Report to National Bureau of Standards for Simulations Models," Department of Computer Science, Virginia Tech. (1977).

[NAN84] NANCE, R. E. (1984), "*Model development revisited*" in Proceedings of the 1984 Winter Simulation Conference, Dallas, TX, pp. 75-80999 - 1984.

[NAN93] NANCE, R. E.,*A history of discrete event simulation programming languages.* ACM SIGPLAN Notices, vol 28, No. 3, March, pp.149-175 - 1993.

[PEG94] PEGDEN, C.D, SHANON, R.E. and SADOWSKI, R.P *Arena Professional Edition Reference Guide* Sewicley PA - 1994.

[QUI94] QUINN, Michael J. *Parallel computing : theory and practice.* 2. ed. Singapore McGraw-Hill, 1994.

[SIL96] SILVA, V. *Multicomputador Nó//: Implementação de Primitivas Básicas de Comunicação e Avaliação de Desempenho*, Dissertação de Mestrado, CPGCC, UFSC, 1996.

[STR94] STRICKLAND, S.G. & PHELAN, R.G. *Massively Parallel SIMD Simulation of Markovian Deds: Event & Time Synchronous Methods,* Dept. of Systems Engineering - University of Virginia -1994.

[ZEF95] ZEFERINO, C. A., LUCKE, H.A.H., SILVA, V.A., *Um multicomputador com Sistema experimental de comunicação.* In. VII Simpósio Brasileiro de Arquitetura de Computadores Processamento de Alto Desempenho, Canela - RS, julho de 1995.

[ZEI76] ZEIGLER, B. P., *Theory of Modeling and Simulation*, John Wiley & Sons, Inc., New York, NY 1976.

ANEXO

Validação

Neste anexo procura-se mostrar alguns valores coletados utilizando a simulação do modelo representativo da arquitetura paralela construído com a ferramenta desenvolvida. Os gráficos foram idealizados segundo trabalhos realizados anteriormente [FRE96], [MER96] e [SIL96]. O painel utilizado do Template Paralelo é o Paralelo INT que possui arquitetura com linhas de interrupção.

Nesta primeira Tabela ANEXO-1 faz-se a análise do tempo de processamento. São variados os números de nós de trabalho, realizando apenas uma comunicação entre os nós e deixando fixos as outras entradas do sistema.

Para um programa seqüencial de 550.000 μs com um número de variáveis igual a 10.000, o acréscimo representa quanto tempo o sistema permanece comunicando além do tempo gasto em processamento. Observar na última coluna os valores obtidos em trabalho anterior.

Número de nós de trabalho	Número de variáveis	Tempo de processamento ideal	Tempo de processamento +comunic (*)	Acréscimo	Tempo (*) obtido em [SIL96]
1	10.000	550.000	550.000	-	-
2	5.000	275.000	309.824	11%	308.061
4	2.500	137.500	173.356	21%	177.568
6	1.667	91.667	122.300	25%	não obtido
8	1.250	68.750	92.463	26%	não obtido
10	1.000	55.000	74.632	26%	não obtido

Tabela ANEXO-1- Análise do tempo gasto em comunicações

Fazendo a análise gráfica da Tabela ANEXO-1, observa-se que a diferença entre o amarelo e o púrpura é o chamado *overhead* associado à comunicação.

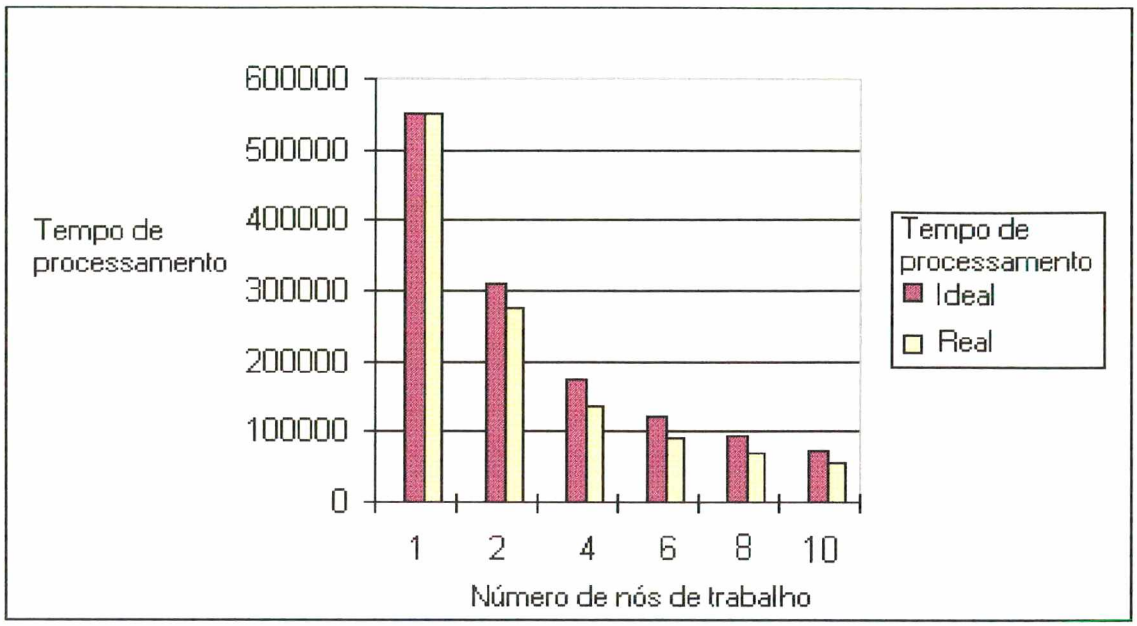


Figura ANEXO-1- Gráfico do Tempo de processamento & Número de nós de trabalho

Um outro ponto que a ferramenta pode explorar é a variação do número de vias de comunicação que interligam o nó de trabalho e o *crossbar*. Este levantamento foi realizado apenas em arquitetura com linhas de interrupção.

Neste experimento é interessante observar que, aumentando o número de vias, ocorre a disputa pelo *crossbar*. Foram fornecidos os seguintes fatores como dados de entrada nas janelas de diálogo do Template:

- Número de nós de trabalho = 10
- Tempo de processamento seqüencial = 550.000 μs
- Tempo de controle = 87 μs
- Capacidade do *crossbar* = 32

- Número de variáveis = 1000
- Número de iterações = 1

Número de comunicações realizadas

	3			4			5		
	tempo(μs)	A	100-A	tempo(μs)	B	100-B	tempo(μs)	C	100-C
1 via	86464	100%		93511	100%		108823	100%	
2 vias	70543	82%	18%	76285	82%	18%	82288	76%	24%
3 vias	65932	76%	24%	69586	74%	26%	73501	68%	32%
4 vias	66891	77%	23%	67324	72%	28%	70725	65%	35%

Tabela ANEXO-2 - Variando número de comunicações e número de vias

A análise da Tabela ANEXO-2 permite mostrar que a variação do número de vias nem sempre melhora a performance do sistema, observar que, com 3 comunicações e 4 vias, ocorreu um aumento do tempo de processamento. Nota-se, pelos relatórios que a ferramenta apresenta, que ocorreu, em certo momento das comunicações, disputa pelo *crossbar*; no caso citado, duas esperas pelo recurso. Porém, no geral, o número de vias melhorou a performance do processamento paralelo em até 35%.

Para a próxima análise foram colocados os seguintes dados de entrada:

- Tempo de processamento = 550.000/número de nós (unidades de tempo)
- Quantidade de vias de comunicação = 1
- Tempo de controle (nó de controle) = 87 (unidades de tempo)
- Permanência no barramento = 3.45 (unidades de tempo)
- Número de variáveis = 10.000/número de nós
- Freqüência de polling = 2 (unidades de tempo)

Na Tabela ANEXO-3 é realizada uma comparação entre as arquiteturas (Interrupção e Barramento de Serviço) realizando uma e duas comunicações.

	Tempo de processamento + comunicação			
Número de	*com Interrup*		*com Barram Serv*	
nós de trabalho	1 comunic	2 comunic	1 comunic	2 comunic
2	309.824	309.824	309.740	309.740
4	173.356	162.421	155.264	173.012
6	122.300	131.249	103.946	127.690
8	92.463	101.187	78.284	92.726
10	74.632	78.273	62.890	78.028

Tabela ANEXO-3 - Comparação entre arquiteturas interrupção & barramento de serviço

A análise desta tabela mostra a arquitetura com barramento de serviço com valores de tempo (de processamento mais comunicação) menores que os valores obtidos na arquitetura que possui linhas de interrupção. A vantagem que a arquitetura com barramento de serviço apresentou, deve-se ao fato do tempo de polling ser muito pequeno (2 unidades de tempo). O barramento de serviço precisa fazer a amostragem das mensagens de comunicação numa unidade de tempo próxima a este valor para obter melhores tempos do que a arquitetura com linhas de interrupção.