B.1.2

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
FACULDADE DE CIÊNCIAS ECONÔMICAS
PROGRAMA DE PÓS-GRADUAÇÃO EM ADMINISTRAÇÃO

MODELO DE SIMULAÇÃO COMPUTACIONAL DE SISTEMAS FLEXÍVEIS DE MANUFATURA

Autor: Denis Borenstein
Orientador: Prof. João Luiz Becker, Ph.D.

Dissertação apresentada ao Programa de Pós-Graduação em Administração como requisito parcial para a obtenção do Título de Mestre em Administração

Porto Alegre, RS
1991

COMISSÃO EXAMINADORA

Prof. João Luiz Becker - Ph. D.
PPGA/UFRGS

Prof. Jaime Evaldo Fensterseifer - Ph. D.
PPGA/UFRGS

Prof. Cláudio Walter - D. Sc.
CPGCC/UFRGS

AGRADECIMENTOS

Ao Prof. João Luiz Becker pelas sugestões dadas durante a realização deste trabalho.

Aos colegas do Departamento de Matemática, Estatística e Computação da Universidade Federal de Pelotas pelo apoio concedido.

Aos funcionários do PPGA-UFRGS pelo carinho dispensado durante a realização do curso.

As bibliotecárias da UFRGS pelo empenho e dedicação com que atendem aos alunos, e cujo auxílio foi fundamental para a realização deste trabalho.

A todos que de uma maneira ou de outra tentaram impedir o término deste trabalho, convidando-me a festas, viagens, etc.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

SIGLAS

ASCII - "American Standard Code for Information Interchange"

AGV - "Automated Guide Vehicles"

ASME - "America Society of Mechanical Engineering"

DNC - "Direct Numerical Control"

EUA - Estados Unidos da América

FMS - "Flexible Manufacturing Systems"

PL - Programação Linear

RAM - "Random-Access Memory"

NOMENCLATURA

BUFFER - armazenador de tarefas em estado passivo, isto é, que não estão sofrendo nenhuma atividade.

DRIVE - mecanismo de acionamento de discos.

JOB-SHOP - sistema de produção em que as facilidades (máquinas) são arranjadas e departamentalizadas segundo um caráter funcional, e que se caracteriza pela capacidade de fabricar uma grande variedade de componentes e produtos.

LAY-OUT - disposição física das facilidades produtivas.

LEAD-TIME - tempo de processamento das tarefas, desde a ordem de fabricação até o término de todas as suas atividades de manufatura.

MACHINING-CENTRE - máquinas capazes de executarem várias operações distintas.

MIX DE PRODUTOS - diversidade de produtos passíveis de serem fabricados por um sistema de manufatura.

PALLET - unidade de padronização de transporte de materiais ou componentes.

PATH - diretório criado pelo usuário em um disco.

SET-UP - tempo requerido para a troca de ferramentas entre duas tarefas distintas em uma máquina.

TRACE DE UM PROGRAMA - acompanhamento da execução, instrução a instrução, de um programa de computador.

TROLLEY - esteira rolante.

WORK-IN-PROCESS - tarefas em processamento, que já tiveram seu processo de manufatura iniciado, porém não concluído.

WORKSTATION - células de manufatura, em que as facilidades são dispostas de uma forma a se constituírem em mini-fábricas.

RESUMO

Este trabalho de pesquisa tem como objetivo desenvolver uma modelagem de Sistemas Flexíveis de Manufatura (FMS), fornecendo a administradores e engenheiros uma ferramenta objetiva de análise a ser utilizada durante o projeto e operação destes sistemas. Essa ferramenta é um modelo genérico de simulação computacional capaz de replicar, em laboratório, os fenômenos que ocorrem no interior de sistemas discretos de produção em lotes. Desta forma, permite a esses profissionais que conheçam e questionem os mesmos, reduzindo os riscos e incertezas inerentes ao processo de implementação prática.

O trabalho apresenta uma discussão do conceito de FMS, explicitando a necessidade de uma ferramenta de análise, bem como uma descrição completa do modelo abstrato desenvolvido e de sua representação computacional, denominada de SIMFLEX.

ABSTRACT

The aim of this reseach is to model a Flexible Manufacturing Systems(FMS), offering to managers and engineers an objective tool of analysis to be used during the design and operation of these systems. The tool is a computational simulation model that emulates, in laboratory, the events presented in batch production systems. It allows a better knowledge and inquiring of these systems by these professionals, decreasing risks and uncertanties inherent in theirs implementation process.

The work presents a discussion of the FMS concept, emphasizing the need of a simulation tool such as this. A complete description of the abstract model developed and its computational representation, called SIMFLEX, is also presented.

1. INTRODUÇÃO

A introdução da automação na área de manufatura, incluindo a automação de informações e do fluxo de materiais, teve como maior conseqüência o aumento da complexidade dos sistemas de manufatura. Em particular, a produção de lotes de pequeno tamanho é bastante complexa, acarretando que as questões ligadas com a automação sejam combinadas com a necessidade por flexibilidade. Desta forma, este tipo de produção tem despertado a atenção dos especialistas nesta área, dando origem ao desenvolvimento de sistemas de produção que agregam tecnologia de ponta, como robôs e sistemas automatizados de transferência de materiais, com grande ênfase na automação e informatização da produção.

Sistemas Flexíveis de Manufatura (do inglês "Flexible Manufacturing Systems", FMS) são sistemas integrados de automação de equipamentos e do fluxo de informações, arranjados de forma a permitir a produção, de forma econômica, de um grupo de componentes complexos em lotes os menores possíveis. Basicamente, estes sistemas são JOB-SHOPs automatizados, consistindo de estações de trabalho e um sistema de transferência de materiais, cujo controle das operações é realizado por um computador central. Contudo, a dificuldade de planejamento destes sistemas, decorrente da integração em um único sistema de elementos sofisticados, tornam a aplicação prática deste conceito um desafio de grandes riscos.

O planejamento destes sistemas é um processo de caráter combinatorial, em que as abordagens otimizantes são freqüentemente impraticáveis, pela dificuldade de solução computacional do modelo formulado (WARNECKE & GERICKE, 1977). A

natureza do problema faz a simulação a única técnica disponível para a modelagem destes sistemas, pois nesta técnica existe uma relação direta entre a seqüência de eventos no sistema real e a seqüência de eventos no modelo.

O objetivo deste trabalho é desenvolver uma metodologia capaz de representar este tipo de sistemas permitindo sua análise durante a fase de planejamento pela estimativa de parâmetros de desempenho. Desta forma será fornecido ao analista informações úteis ao seu processo de tomada de decisão no sentido de escolher a configuração geral (máquinas, "mix" de produtos etc.) que melhor se adequa aos seus critérios, previamente definidos. Neste sentido, foi construído um modelo genérico de simulação computacional, que replicará em laboratório os fenômenos que ocorrem em um FMS.

A fim de facilitar a leitura deste trabalho, o mesmo foi dividido nas seguintes seções:

1a. - Capítulos 2 e 3. Apresentam e contextualizam o conceito de FMS no atual panorama da engenharia de produção, bem como discutem suas vantagens e os problemas que ocorrem entre sua concepção teórica e sua implementação prática.

2a. - Capítulo 4. Apresenta os métodos quantitativos aplicáveis a uma modelagem de FMS, justificando a simulação como técnica adequada.

3a. - Capítulos 5 e 6. Apresentam os objetivos do trabalho, gerais e específicos, e a metodologia desenvolvida para que os mesmos fossem alcançados.

4a. - Capítulo 7. Descreve a estrutura do modelo genérico, apresentando uma definição operacional de termos e

variáveis relacionadas às entidades do modelo.

5a. - Capítulo 8. Descreve o sistema computacional SIMFLEX, desenvolvido em linguagem PASCAL, utilizando o compilador TURBO PASCAL 5.0, compatível com microcomputadores da linha IBM-PC. É apresentado uma explanação dos vários módulos que compõem o programa, bem como orientações práticas para a sua utilização, e o seu processo de verificação.

6a. -Capítulo 9. Apresenta uma avaliação crítica do processo de modelagem realizado e de sua programação, assim como uma série de sugestões para futuras pesquisas.

2. CONTEXTUALIZAÇÃO DOS FMS DENTRO DOS NOVOS CONCEITOS DE ORGANIZAÇÃO DA PRODUÇÃO

A expansão da produção de bens industriais ocorrida em vários países, inclusive naqueles sem tradição industrial, durante a década de 50, apresentou como principal decorrência uma alteração nas leis de formação destes mercados a partir da década de 60. Esta alteração caracterizou-se pelo excesso de capacidade instalada, fazendo com que a oferta pudesse superar a demanda (CORIAT, 1985). A principal estratégia até então utilizada, de produzir grandes quantidades e volumes ao menor custo passou a não mais ser aplicada, sendo substituída pelo conceito de qualidade. Este conceito foi introduzido por novos competidores para se diferenciarem dos tradicionais fabricantes, sendo priorizado como o elemento estratégico de produção capaz de ganhar mercados . Este fato colocou em crise definitiva a forma reinante de organização de trabalho, traduzida pelos princípios tayloristas/fordistas, e popularmente denominada como produção em massa: a produção de grandes volumes de bens, com o objetivo de minimizar o custo unitário de produção.

Esta organização de trabalho baseia-se sobretudo na rigidez dos sistemas produtivos, através da busca de uma uniformização da produção, quer nas formas de utilização dos dispositivos técnicos, vinculando-os às características do produto, quer no uso da força de trabalho.

A nova formação dos mercados começou a exigir novas formas de organização do trabalho, necessárias para aumentar a eficiência e eficácia da produção. Esta nova concepção deu origem a duas correntes de adaptação às novas características do

mercado, que podem ser desta forma resumidas (SALERNO, 1987):

Primeira - **Baseada na Intensificação do Trabalho**, ou seja, a partir de uma tecnologia constante (ou com baixos índices de inovação), com um mesmo número de trabalhadores se consegue produzir no mesmo tempo uma quantidade maior de produtos.

Para que isto possa se tornar possível é necessário uma completa reformulação na organização da produção, eliminando tempos mortos, ou seja, quaisquer tempos que não estejam diretamente ligados ao acréscimo de valor ao produto.

Esta foi a filosofia adotada pelos japoneses iniciando com os famosos Círculos de Controle de Qualidade (CCQ) até a introdução de técnicas não informatizadas de controle de gestão da produção, tipo Kanban.

Toda a operacionalização desta filosofia é dirigida ao mercado, concedendo a necessária flexibilidade às linhas de produção a fim de possibilitar uma integração com o mesmo.

Segunda - **Baseadas no Aumento de Produtividade Técnica**, ou seja, dentro de um mesmo ritmo de trabalho, e com a mesma quantidade (ou de preferência com menor número) de trabalhadores, se obter uma produção maior, conseqüente da maior eficiência técnica dos meios de produção.

Esta filosofia predominou no mundo ocidental, principalmente nos EUA, e se utilizou da automação como a arma mestra para atingir a integração com o mercado, originando linhas de produção automatizadas de forma rígida.

É inegável o sucesso da primeira corrente em relação à segunda, bastando para tal observar a situação da economia japonesa dentro do contexto mundial. A principal razão para tal se encontra no fato de que a primeira teve como principal objetivo a busca de uma nova organização da produção em detrimento da introdução súbita de inovações tecnológicas. Caminho oposto foi seguido pela segunda, em que se implementaram inovações tecnológicas, de alto custo, em uma estrutura totalmente inadequada ao mercado, acreditando-se que a automação per se seria capaz de alterar o status quo da estrutura do chão-de-fábrica. Altos investimentos foram feitos pelas indústrias ocidentais, encontrando pela frente produtos de alta qualidade e de baixo preço fabricados pelos japoneses, causando imensos prejuízos.

Desta maneira, começou a se observar que o grau de sofisticação não é um critério decisivo. O determinante chave das relações de competitividade entre as empresas é, na verdade, a capacidade de inovação e de criatividade desenvolvidas nas linhas de produção, e na capacidade de desenvolver soluções adequadas a cada situação particular de uma ou de várias novas propriedades oferecidas pela série de equipamentos eletrônicos.

2.1 Flexibilidade e Integração: Conceitos Fundamentais dentro da Atual Engenharia de Produção

Como decorrência destas profundas transformações surge a necessidade da formação de novos sistemas de produção, que venham ao encontro das novas características dos mercados. Estes sistemas devem ser orientados por dois novos conceitos: flexibilidade e integração.

2.1.1 Flexibilidade

Segundo SALERNO (1987) : "Flexibilidade aparece como um atributo indissociavelmente ligado à automação microeletrônica; o senso comum a respeito considera flexibilidade como a capacidade de mudar rapidamente o produto em fabricação" .

A flexibilidade estaria assim, após a consolidação da qualidade, tornando-se o novo conceito estratégico capaz de dotar sistemas de produção do elemento diferenciador necessário para sobrepujar competidores.

Dentre as várias definições de flexibilidade existentes na literatura, escolhemos aquela de caráter mais geral, segundo a qual a noção de flexibilidade cobriria as seguintes categorias (CORIAT, 1985):

- flexibilidade do "mix" de produtos - possibilidade de fabricar simultaneamente um conjunto de produtos com características de base comum;
- flexibilidade de mudança de projeto - capacidade de modificar rapidamente o processo, mudando as características a serem dadas às peças;
- flexibilidade de peças - possibilidade de acrescentar ou suprimir uma peça do processo;

- flexibilidade de volume - capacidade do sistema de adaptar-se às flutuações de volume da produção de uma peça, modificando os ritmos e os tempos de operação e de ocupação das ferramentas;
- flexibilidade de roteamento - dada uma situação de máquina bloqueada, em pane ou saturada, o sistema automaticamente tem a capacidade de redirecionar uma peça para uma outra máquina, que tenha um espaço livre de trabalho e esteja pronta para ser acionada.

2.1.2 Integração

Dentro do escopo deste trabalho, integração é a idéia que contém dentro de seu âmbito a noção de que a eficácia do conjunto não é igual à soma da eficácia de seus componentes. Em outras palavras, a otimização do todo não será obtida através da otimização particular de suas partes (CORIAT, 1985). Abordando-se a integração desta forma, estabelece-se para a mesma um significado global, que servirá desde o chão de fábrica até as relações empresa-sociedade.

a) A nível de chão de fábrica. A integração está diretamente relacionada à simplificação do processo de manufatura, através da modularização e formalização de todas as tarefas executadas. A partir deste fato, as seguintes vantagens seriam obtidas:
   - melhor uso dos recursos;
   - minimização dos custos de produção;
   - redução do "lead-time", pelo melhor aproveitamento dos tempos de circulação de materiais e de operação;

b) A nível organizacional. Deve-se salientar que a área produtiva não é isolada, ao contrário, ela faz parte de um sistema maior e mais complexo, a organização, que necessariamente interage com quase todas os outros sub-sistemas que a compõem. Desta forma não se pode pensar na produção como uma ilha isolada, mas como um sub-sistema em constante integração com outros, como compras, finanças, distribuição etc.

Uma implicação imediata é que não se pode ter uma linha de produção altamente sofisticada se os demais sub-sistemas da empresa não se adequam ao mesmo, traduzindo este fato em completo desperdício de capital.

c) A nível empresa-sociedade. A organização não é um sistema fechado, ela está em constante interação com o ambiente, seja na troca direta de produtos/serviços - capital, seja no relacionamento político-social entre ela e a sociedade.

Neste sentido deve-se buscar uma integração entre a organização e a sociedade, a fim de que aquela se amolde às necessidades desta, de maneira a só produzir na qualidade e quantidade desejadas pelo mercado, evitando-se assim a fabricação de produtos sem perspectivas de venda.

A informação será a ferramenta básica capaz de realizar o objetivo mestre da integração, e a sua capacidade de manipulação de forma eficaz e eficiente o requisito fundamental para a sua obtenção. Ressalta-se que a microeletrônica parece ter dado grande alento a estes objetivos, permitindo a informatização da

gestão empresarial. Porém, as grandes variações e a dificuldade de manipulação computacional de todas as informações, principalmente informais, tornam esta meta um desafio de grandes proporções.

3. SISTEMA FLEXÍVEIS DE MANUFATURA (FMS)

Conforme a automação e a informatização vêm se disseminando na área de manufatura, é natural o surgimento de sistemas produtivos que delas se utilizam para atingir o objetivo de se compatibilizarem com o mercado, principalmente na fabricação de lotes de médio e baixo volumes. O primeiro destes sistemas foi a linha rígida automatizada, que basicamente é uma linha de montagem no estilo fordista, em que operadores humanos são substituídos por máquinas automatizadas. Inobstante, o grande desenvolvimento da microeletrônica possibilitou o surgimento de sistemas mais adequados às exigências da nova estrutura de mercados. Neste sentido, pode-se considerar o FMS o estado atual da arte em termos de sistemas de manufatura automatizados e informatizados.

3.1 Célula de Manufatura

O conceito de FMS começou a ser desenvolvido na Universidade de Trondheim, Noruega, através de um outro conceito, de células de manufatura (KOREM, 1982). Para melhor percepção deste conceito é necessário entender alguns aspectos sócio-econômicos da Noruega. Este país era composto por uma população pequena e dispersa, característica desejável por questões de segurança nacional. Problemas econômicos nas áreas rurais levaram grande massa de desempregados a procurar emprego nas indústrias instaladas nos maiores centros. Por razões econômicas e de segurança nacional, o governo norueguês contratou os serviços daquela universidade para evitar este deslocamento populacional

do campo para a cidade. A solução encontrada foi a de projetar pequenas mini-fábricas de manufatura para as áreas rurais.

Neste conceito celular, as operações de manufatura são quebradas em células, cada uma podendo ser considerada como uma fábrica. Cada célula é responsável pela manufatura de uma específica família de peças, determinada pelos princípios de tecnologia de grupo. As células são interconectadas por uma rede de materiais e transporte de componentes.

Após a consolidação desta fase em que os objetivos iniciais foram totalmente obtidos, começou a ser observado que em função dos altos salários noruegueses e a fim de tornar a indústria competitiva dentro do mercado mundial, a produtividade destes sistemas deveria ser extremamente alta. Estes fatores conduziram ao segundo conceito no plano: prover cada mini-fábrica com a mais alta tecnologia disponível.

Assim, uma célula passou a se constituir de um grupo de várias (duas a cinco) máquinas de controle numérico arranjadas em círculo em torno de um robô, como exemplificado na figura 1. O robô realiza todas as operações de transporte dos componentes e materiais e a preparação das máquinas para as operações necessárias. A execução e coordenação das várias operações é realizada por um computador celular, o qual poderá ou não estar ligado a um computador central. Este conjunto pode funcionar 24 horas, e trabalhadores serão necessários somente a fim de realizar as seguintes tarefas: programação de computadores, planejamento da produção, escalonamento, manutenção, tratamento térmico dos componentes necessários, montagem dos materiais nos sistemas de transporte, etc.

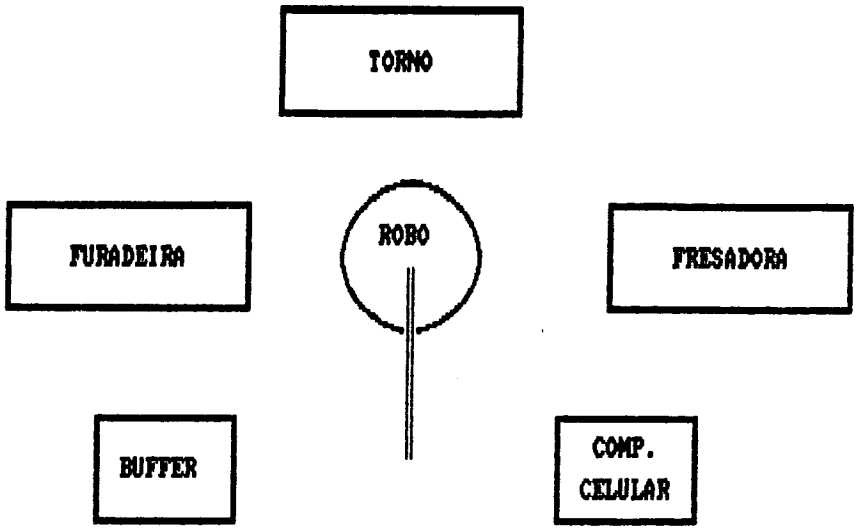


Figura 1 - Arranjo típico de uma célula de manufatura

A otimização da taxa de produção de uma célula de manufatura requer uma estratégia diferente daquela aplicada para um centro de trabalho convencional. Uma boa estratégia de otimização de uma célula é a de equalizar o tempo de utilização de cada máquina operatriz. Com esta estratégia o tempo de ciclo de produção de uma célula é igual ao tempo de utilização da máquina mais lenta, acrescido da porção de tempo relativo a mudança de ferramenta ("set-up"), e do tempo de carregamento e descarregamento dos componentes ou materiais.

3.2 Conceito de FMS e seus Princípios

A incorporação de várias células de manufatura, como as descritas acima, em um grande sistema, na qual a produção de componentes é controlada por um computador central, pode ser considerado como o conceito básico de um FMS. Esta designação é proveniente da alta flexibilidade provida por este sistema, em

termos do pequeno esforço e reduzido tempo requerido para a manufatura de um novo produto.

Desta forma, o FMS pode ser considerado como um "job-shop" automatizado. Conseqüentemente apresenta todos os problemas tradicionais associados ao controle destes: escalonamento de tarefas, nível de utilização de capital e desempenho em termos de produtividade ("lead-time", "work-in-process", etc.).As principais diferenças entre "job-shops" convencionais e o conceito de FMS são:

- as funções humanas são automatizadas, inclusive às relacionadas com tomadas de decisões;
- as máquinas apresentam uma disposição celular;
- os tempos das atividades baixam de meses ou semanas para horas.

Basicamente, um FMS será composto dos seguintes princípios (CORIAT, 1985):

a) Em relação à fabricação - baseado em células totalmente automatizadas, denominadas de estações de trabalho ("workstations"). As células apresentadas na figura 1, ao incorporarem importantes avanços tecnológicos, deram origem a células compostas de máquinas multi-funcionais, isto é, capazes de executarem várias atividades de usinagem, e denominadas de "machining-centre". Estas máquinas utilizam a tecnologia de controle numérico direto ("DNC", do inglês "Direct Numerical Control"), e agregam manipuladores capazes de realizar as funções exercidas pelo robô na figura 1. Desta forma, estas estações de trabalho possuem tecnologia para realizar,

com menor espaço e com menor número de equipamentos, as atividades realizadas pelas células anteriormente descritas (HUTCHINSON, 1980 e YOSHKAWA, 1980).

Estes "machining-centres" deverão agregar as seguintes características (COLLINS, 1980):

- Uma grande capacidade de movimentação da ferramenta;
- Um acionamento extremamente flexível em termos de velocidades e torque, a fim de suportar um grande número de operações e de materiais;
- Facilidade de acesso para limpeza e manutenção.

b) Em relação à circulação de materiais - ocorre através de sistemas de transferência automatizados, entre os quais destacam-se:

- "trolleys" automáticos;
- correias transportadoras;
- AGV (do inglês "automated guided vehicles").

c) Em relação à integração com o chão de fábrica - a gestão é realizada de forma informatizada, por um computador central, de tal sorte que quaisquer alterações no fluxo de produção são realizadas em tempo real.

O computador central, desta forma, é responsável por duas funções:

i. - Controle das tarefas programadas;

ii.-Controle da utilização da maquinaria.

É importante ressaltar que deverá existir uma compatibilidade entre o computador central e os seus vários periféricos (centros de trabalho e sistema de transferência), para que estas funções sejam executadas

de forma eficaz.

d) Em relação à flexibilidade - como os recursos são programáveis, com capacidade de armazenar vários programas de diferentes operações, os mesmos são capazes de reconhecer, em tempo real, a partir de uma ordem qualquer de peças em fila, o programa de operações adequado a cada peça diferente.

Desta forma, os sistemas flexíveis incorporam de forma integral o conceito de flexibilidade descrito no item 2, e a nível de chão de fábrica o conceito de integração. Os FMS vêm assim ao encontro aos anseios do mercado, permitindo, de forma econômica, a produção de uma grande variedade de produtos em pequenos lotes.

3.3 Desenvolvimento de um FMS

O projeto e aquisição de um FMS apresenta as seguintes fases de desenvolvimento (SURI, 1985), comuns a qualquer sistema automatizado:

1a. Fase - **Concepção do Sistema**. Esta fase envolve um estudo de viabilidade ou um projeto preliminar do sistema, com o objetivo primário de estabelecer a adequabilidade do mesmo, com a determinação de certos parâmetros, como a estimativa de aspectos estratégicos, financeiros, logísticos e operacionais. Como resultado desta fase, algumas alternativas devem ser levantadas.

2a. Fase - **Processo de Planejamento e Configuração do Sistema**. Nesta fase, cada uma das alternativas

selecionadas na primeira fase deve ser cuidadosamente analisada; as características do sistema devem ser identificadas e um projeto detalhado deve ser executado em termos de seleção de equipamentos e "lay-out" das estações de trabalho.

3a. Fase - **Instalação.** Inclue o processo de instalação do sistema e determinação de procedimentos para sua realização.

4a. Fase - **Operacionalização.** Após sua instalação, o sistema deve ser operado no sentido de atingir os objetivos estabelecidos nas fases 1 e 2.

5a. Fase - **Avaliação.** Mesmo após o sistema já estar totalmente operacionalizado, existem questões de longo-termo que devem ser avaliadas, no dia-a-dia de operação, a fim de melhorar o sistema e adequá-lo às modificações no mercado.

3.4 Situação Atual dos Sistemas Flexiveis de Manufatura

De acordo com TCHIJOV & SHEININ(1989) o conceito de FMS tem sido aplicado a uma grande variedade de sistemas de produção em lote, e nas mais diversas áreas da indústria metal-mecânica.Os autores estimam que existam cerca de 500 FMS em operação no mundo no final de 1986, espalhados pelos países da Europa, EUA e Japão.

A literatura sobre automação industrial apresenta descrições dstes sistemas, geralmente limitando-as a detalhes técnicos e dados operacionais, que mencionam benefícios em relação a sistemas convencionais, embora o número de dados disponíveis seja

bastante limitado. Isto é ratificado pelos dados apresentados por TCHIJOV & SHEININ(1989) ao concluir, após analisar cerca de 400 sistemas, que somente 11.25% possuem dados confiáveis sobre a redução no "lead-time". O mesmo ocorre para os demais parâmetros de desempenho: 6.5% casos para redução no tempo de "set-up", e 8.25% casos para a redução em tempo de processamento.

Uma detalhada análise de 119 sistemas realizada por YTO(1982), concluiu que somente seis dos mesmos englobavam em sua totalidade os cinco níveis de flexibilidade citados em 2.1.1. A maior parte dos sistemas estudados trabalham com um número limitado de "mix" de produtos - freqüentemente tão baixo como dois ou três. O panorama se apresenta mais alentador na pesquisa de TCHIJOV & SHEININ (1989), onde em 222 casos levantados, a média aumentou para 14 produtos. Em 57% destes casos, a flexibilidade é representada por menos de 30 produtos, sendo que um terço destes FMS produzem menos de 10 produtos diferentes.

Estes dados são bastante importantes para se afastar a idéia de que todos os sistemas existentes suportam a produção de uma grande variedade de componentes ou produtos limitados somente pelas restrições de peso e dimensões projetadas para o sistema.

Aliando estas informações à constatação de que este conceito já existe há cerca de 30 anos (INFOTECH, 1980), e que as barreiras tecnológicas já podem ser consideradas vencidas a partir da afirmação de 90% dos engenheiros relacionados com automação interrogados pela ASME em 1978 (COLLINS, 1980): "que em 1985 já existiria a necessária tecnologia para suportar um sistema de produção como o FMS", conclui-se que este tipo de sistema apresenta problemas com relação ao seu real desempenho,

especialmente em áreas que exigem alto grau de sofisticação. Existe uma barreira entre a concepção teórica e a sua operacionalização, ficando esta bem aquém do esperado, colocando em dúvida às vantagens apregoadas. Estes problemas são decorrentes dos dois fatores abaixo analisados.

a)**Dificuldade de percepção dos conceitos da nova engenharia produtiva existentes dentro do âmbito dos FMS**

A percepção e entendimento dos novos conceitos oriundos da nova formação dos mercados se realizam, geralmente, dentro de um processo caracterizado por grandes problemas, motivado pelo conservadorismo existente na área de produção. Deste modo, a grande maioria das modernizações industriais têm sido realizada pela simples substituição de equipamentos, sem que haja um questionamento da organização como um todo. Permanecem assim na estrutura produtiva os mesmos vícios da organização taylorista, ou seja, a produção de bens a altos volumes, minimizando os custos unitários, e que não tiram proveito do maior benefício trazido por sistemas complexos como o FMS: a flexibilidade.

Segundo MEREDITH & HILL(1987) uma das grandes causas para a falha de sistemas avançados de manufatura é a falta de um total entendimento da tecnologia envolvida antes da mesma ser implementada. Estes sistemas complexos demandam um extenso pré-planejamento, onde não só a tecnologia a ser empregada deve ser analisada ao extremo, mas também os seus efeitos na organização como um todo. Assim, o grau de inovação tecnológica não é o critério decisivo, mas sim a

capacidade de inovação e de criativiodade desenvolvidas na produção, que só serão plenamente obtidas com a perfeita percepção dos conceitos já apresentados como fundamentais dentro do novo contexto em que se encontra a engenharia produtiva: flexibilidade e integração.

b)Métodos Econômicos de Análise que Justifiquem sua Implantação

A análise econômica de um FMS é bastante difícil de ser realizada. Os argumentos financeiros estão baseados em fatores que não levam em conta parâmetros de extrema relevância, priorizando os de mais rápida identifição. Métodos econômicos disponíveis, como custo-benefício, mostram-se inadequados a uma análise sobre as vantagens da implantação de um sistema desta natureza, pois a quantificação dos custos é relativamente mais fácil de ser acessada em relação a quantificação dos benefícios, em especial os intangíveis como a flexibilidade.

De acordo com MEREDITH & HILL (1987), é imprópria a utilização de justificativas financeiros de análise, baseados sobretudo na redução de custos, em decisões que envolvam aspectos estratégicos, como é o caso dos FMS. Estes aspectos, por suas características, exigem análise de longo-termo, fato que os métodos econômicos convencionais não conseguem captar. HASEGAWA (1985) percebendo estas dificuldades, expõe a necessidade da utilização para uma análise econômica de sistemas automatizados de métodos de análise multi-critérios, como Análise de Valor e Análise de

Risco, que são extremamente complexos e exigem um conhecimento representativo do sistema a ser estudado.

É inegavel assim que os fatores acima citados são os principais limitantes para a expansão do conceito de FMS, e a busca de uma ferramenta de análise condição fundamental para a sua expansão.

4. MODELAGEM DE SISTEMAS

Três características estão fortemente associadas ao desenvolvimento de um FMS, quais sejam :

a. - Altos investimentos de capital;

b. - Alta complexidade do sistema (gerando dificuldade de entendimento dos conceitos agregados);

c. - Altos riscos devido à natureza da tecnologia;

Estas características podem ser consideradas como aspectos centrais para a justificativa da construção de modelos, com o objetivo de acessá-las e permitir uma ferramenta de análise, capaz de fornecer dados de desempenho do sistema, durante o seu planejamento. Conseqüentemente, diminuindo as incertezas decorrentes das mesmas.

Os modelos se constituirão em uma forma de representação do sistema que permitirá que se manipulem e compreendam as entidades envolvidas, tanto em seus aspectos qualitativos como em seus aspectos quantitativos (STRACK, 1984). Desta forma, os modelos, no âmbito de suas formulações, forneceriam os seguintes aspectos, que os colocariam como excelente resposta aos três problemas acima referenciados:

a. - A necessidade de organização, avaliação e validação das idéias existentes no sistema, para a sua representação de forma clara e objetiva. Neste sentido, conceitos são ordenados e inconsistências eliminadas (STRACK, 1984).

b.- A realização de previsões, permitindo estudos antecipados de comportamentos e situações, podendo as mesmas serem detectadas antes de sua ocorrência.

4.1 Visão Geral dos Métodos Quantitativos Aplicáveis

O esquema apresentado na tabela 1 apresenta os métodos quantitativos que podem ser aplicados a uma análise sobre sistemas com as características dos FMS (SURI, 1985). Deve-se ressaltar que esta classificação tem como principal objetivo fornecer um panorama geral dos métodos, não apresentando rigor, pois as fronteiras criadas não são perfeitamente definidas. Por exemplo, existem algumas modelagens dinâmicas de Programação Linear (PL), embora a grande maioria de aplicações de PL sejam estáticas. A inclusão da tabela neste trabalho é justificada pelo seu potencial didático.

O esquema é baseado em dois atributos. O primeiro questiona se o método é baseado em modelos determinísticos ou probabilísticos. Neste as incertezas da situação a ser estudada estão agregadas dentro do modelo, enquanto para os determinísticos estas incertezas são negligenciadas ou modeladas indiretamente (através de fatores de segurança). O segundo atributo é se o método evolui e os componentes do sistema interagem com o tempo (dinâmicos), ou o comportamento do sistema é analisado de maneira sumarizada dentro do período total de tempo a ser estudado (estáticos).

Deve-se observar que nesta categorização a acurácia cresce para a direita e para baixo. Isto não quer dizer que o modelo mais preciso seja sempre o melhor. Deverá haver um compromisso entre o que se deseja e a precisão necessária, pois o tempo de desenvolvimento e o de utilização de recursos computacionais aumentam geralmente com a acurácia.

	Atributo 2	
Atributo 1	Estáticos	Dinâmicos
Determinísticos	Programação Linear	Programação Dinâmica
Probabilísticos	"Queueing Network"	Simulação

Tab. 1 - Categorização dos Métodos Quantitativos

4.2 Simulação como Técnica Adequada

A natureza dos processos industriais em uma produção por lotes, onde existe uma grande variabilidade das variáveis e um grande inter-relacionamento entre as mesmas, faz com que a representação de sistemas, como os FMS por modelos determinísticos se torne impossível e inválido (WARNECK & GERICK 1977).

Já os "Queueing-Network Models" fornecem modelos simplistas de um sistema real. Tais modelos apresentam como principal característica a forma generalista em que são construídos, que em conjunção com a simplicidade, proporcionam modelos rápidos e baratos de usar. Recomenda-se sua utilização durante a elaboração da Fase de Concepção do Sistema (ver seção 3.3).

Contudo, ao se avançar para a 2a. fase de desenvolvimento do

sistema, de Planejamento e Configuração do Sistema, onde devem ser incorporados os acontecimentos que se observam no dia-a-dia de um ambiente industrial, como atrasos na entrega de materiais, manutenção, falhas nos equipamentos, etc., estes modelos rapidamente explodem, tornando-se abstrações não-válidas da realidade, e conseqüentemente não fornecendo soluções adequadas para os problemas apresentados na seção 3.4.

É necessário, portanto, que se encontrem modelos mais detalhados e poderosos, capazes de representar a estrutura complexa dos processos industriais, que ocorrem de maneira discreta, através de eventos e da relação entre os mesmos. Estes modelos demandam um considerável entendimento do sistema a ser estudado. São caros, pois consomem mais tempo, pessoal especializado e recursos computacionais. Estas representações caras e sofisticadas recebem o nome de simulação (RATHMILL, 1980).

Existe um consenso na literatura que simulação é a ferramenta adequada caso um modelo detalhado deva ser construído (SARGENT, 1988). No início de 1979, em um seminário em Manchester, Dr. Eugene Merchant respondendo a uma questão relacionada com a justificativa financeira de tais sistemas, afirmou categoricamente : "simulação computacional é a chave para acessar desempenho e minimizar riscos" (RATHMILL, 1980).

Dentro do âmbito da simulação, existe uma relação direta entre a seqüência de eventos no sistema real no tempo real e a seqüência de eventos no modelo computacional no tempo de modelagem. Deve-se ressaltar que a simulação, ao contrário da modelagem matemática, não fornece uma solução ótima, mas uma

análise de dados operacionais de desempenho de um sistema sobre um conjunto de condições, tão variadas quanto se desejar. Um importante aspecto na modelagem por simulação é a necessidade que ela requer de uma manipulação inteligente de todos os processos de projeto do sistema a ser estudado. De maneira que, para o caso de um FMS, as várias opções e variações existentes deverão ser avaliadas e, onde possível, eliminadas em uma série de investigações. Freqüentemente, as fases iniciais de modelagem já fornecem uma razoável lista de variáveis a serem examinadas; e quando o modelo detalhado é construído cada variável poder ser, então, investigada.

Uma discussão completa dos conceitos existentes no âmbito da simulação pode ser encontrada em LAW & KELTON (1982).

5. OBJETIVOS

O objetivo deste trabalho é a de idealizar uma representação de FMS, sistemas consistindo de estações de trabalho automatizadas (denominadas de células) interligadas por um sistema de transferência de materiais, cujo controle das operações é realizado por um computador central, fornecendo uma ferramenta objetiva de análise a ser usada, nas fases de planejamento e configuração destes sistemas.

Além deste objetivo geral, a representação deverá fornecer subsídios úteis para a compreensão dos seguintes aspectos gerais de um sistema de produção em lotes (HUGHES, 1977):

- conceitos básicos do sistema (flexibilidade, em especial);
- fluxo de materiais;
- algoritmos de escalonamento,

e soluções para questões específicas, como:

- número e tipo de máquinas operatrizes em uma estação de trabalho;
- número máximo de itens no sistema;
- quebras na maquinaria e nos sistemas de transferência de materiais.

6 METODOLOGIA

Para que os objetivos desta pesquisa sejam alcançados será utilizada uma metodologia experimental, pelo desenvolvimento de um modelo genérico de simulação computacional que replicará, em laboratório, os fenômenos que ocorrem em um FMS, descrito de acordo com a seção 3 (CERVO & BERVIAN, 1978).

Este modelo genérico, que pode ser considerado como um protótipo paramétrico de simulação de FMS, permitirá observações sobre os processos que ocorrem nestes sistemas de produção, antes e durante sua implementação, questionando a sua natureza, assim como criará condições para a sua variabilidade (em número de células, máquinas etc.) e repetição, para melhor ser observado em suas causas, conseqüências e desempenho.

7. MODELO DE SIMULAÇÃO COMPUTACIONAL

7.1 Requisitos

No capítulo 3 foi apresentada a dificuldade de se analisar, de maneira objetiva, os benefícios trazidos por um sistema complexo como o FMS, e portanto de justificar a sua implementação fora do âmbito tecnológico. O modelo de simulação computacional a ser apresentado neste trabalho foi desenvolvido com o intuito de ser um avaliador da potencialidade de um FMS, constituindo-se em um modelo abstrato sobre o qual o analista modelará o seu sistema.

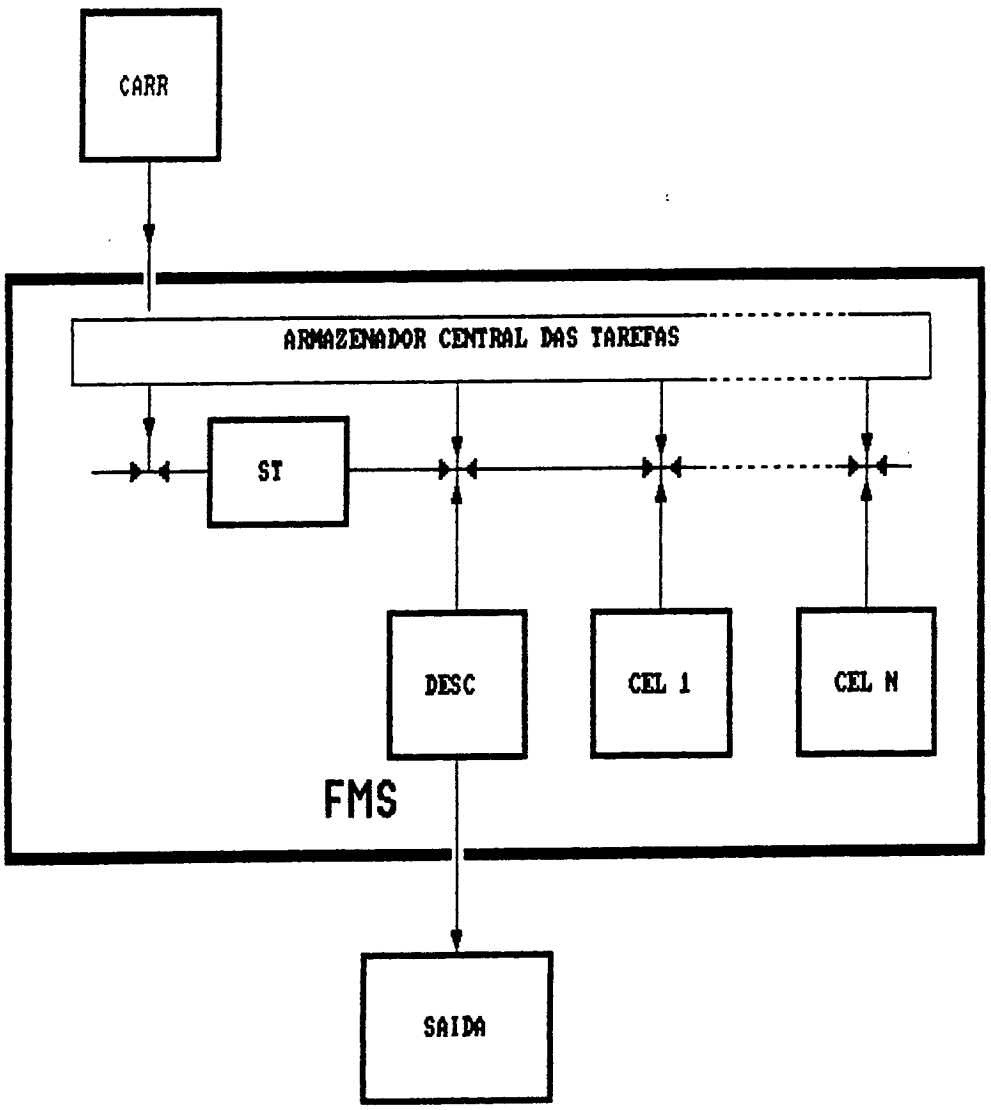
Desta forma, o modelo de simulação computacional deverá apresentar como atributos os seguintes aspectos:

- permitir ao usuário a modelagem de inúmeros sistemas distintos, tanto no seu "lay-out" como no "mix" de produtos, oferecendo um modelo flexível para um sistema que oferece este conceito como sua maior vantagem;
- resultados que permitam uma análise precisa do sistema analisado;
- facilidade de manipulação pelo usuário, de forma a permitir seu uso por pessoal que desconheça técnicas refinadas de programação;

7.2 Descrição do Modelo de Simulação

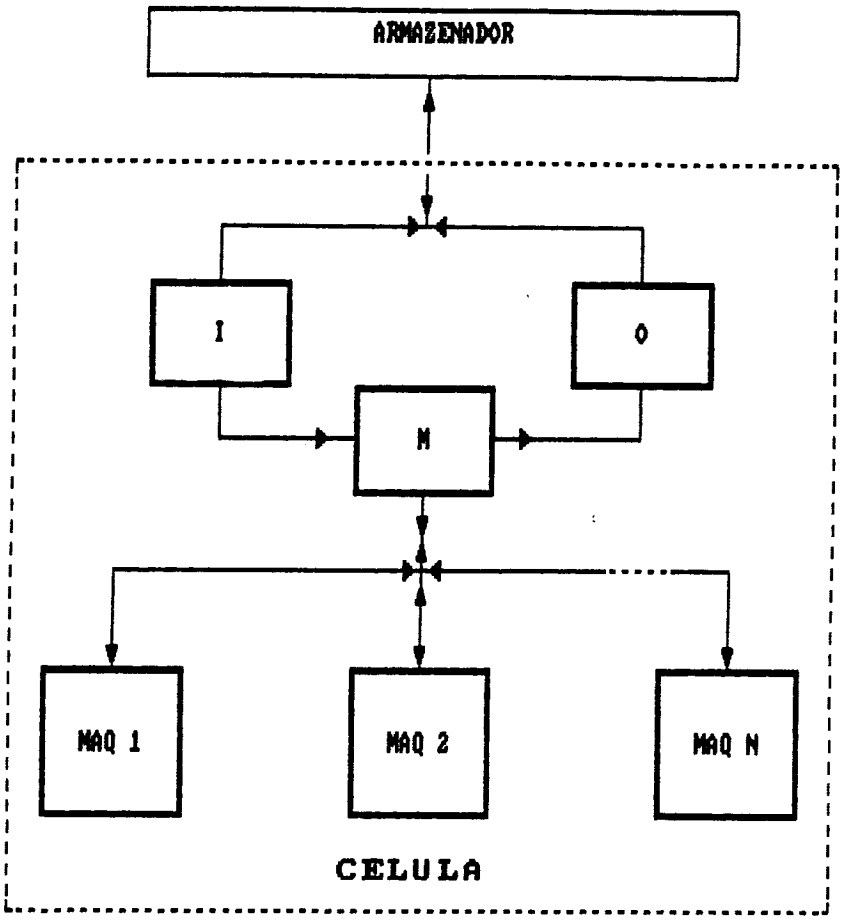
A figura 2 apresenta a estrutura do modelo abstrato a ser utilizada na definição do simulador.O mesmo pode ser considerado como uma extensão do modelo apresentado por WARNECKE & GERICK (1977).

As entidades carregamento e saída, que possuem como função respectivamente alimentar o sistema com matérias-primas e armazenar as tarefas que acabaram todas as atividades de manufatura, apresentam um caráter eminentemente logístico, não participando de atividades de manufatura. Deste modo, estas entidades se encontram fora do âmbito de abrangência do modelo, como pode se observar na figura 2.



ST = sistema de transferencia
CARR = carregador do armazenador central
DESC= descarregador do armazenador central
CEL = celula de trabalho

Fig. 2a - Modelo Abstrato do Sistema FMS



I = input para as celulas
O = output para as celulas
M = manipulador
MAQ = maquinas operatrizes

Fig. 2b - Modelo Abstrato do Sistema FMS (Entidade Célula)

7.2.1 Definição Operacional de Termos e Variáveis

Neste item cada um dos elementos diretamante relacionados com o modelo abstrato apresentado acima será definido.

Célula

Entidade composta de um e somente um manipulador (agregado ou não a uma máquina), e um conjunto de máquinas , diferentes ou não.

Atributos:

- Identificação - variável que diferencia às células entre si;
- Número de Máquinas;

"Buffer" de Entrada/Saída de uma Célula

Entidade responsável pelo armazenamento das tarefas que:

- chegam a uma célula para operação em máquina;
- por motivo de pane ou saturação de máquina necessitam aguardar o restabelecimento da mesma;
- ficam à espera do Sistema de Transferência para ser conduzida a outra célula;

Atributos:

- Localização - coordenadas cartesianas que posicionam o "buffer" no interior do sistema, em relação a um referencial arbitrado pelo usuário.

Manipulador

Entidade responsável pela locomoção das tarefas entre o "buffer" celular e as máquinas, e vice-versa; e pela realização das atividades de "set-up" das máquinas.

Atributos:

- Identificação: variável que associa o manipulador à célula em que se encontra.
- Velocidade: velocidade média de atuação do manipulador, em

unidade compatível com a modelagem realizada.

- Condição de Operação: O manipulador possui dois modos de operação:

- rotina - o manipulador se encontra operando normalmente;

- em pane - o manipulador se encontra avariado, e portanto não realiza nenhuma de suas funções.

Não foram previstas, no caso de se utilizarem manipuladores automáticos (robôs, por exemplo), capacidade de alteração dos meios de fixação dos componentes, garras ou ferramentas especiais, acreditando-se ser razoável que estas atividades sejam realizadas durante a ociosidade ou durante os períodos de transporte, pelo sistema de controle.

As rotas possíveis ao manipulador são fornecidas em termos das distâncias existentes no interior de uma célula, supostamente conhecidas . O tempo requerido para o transporte de um determinado componente é calculado a partir da distância envolvida e da velocidade média do manipulador.

A fim de simplificação do modelo, foi considerado que o sistema da controle será capaz de pré-alocar o manipulador, onde se fizer necessário, um período de tempo antes do mesmo ser efetivamente requisitado. Desta forma, não serão contabilizados atrasos decorrentes da locomoção de manipuladores desocupados.

Máquina

Entidade responsável pelas atividades de transformação, de qualquer natureza, das tarefas em estágios cada vez mais avançados em direção ao produto final.

Atributos:

- Identificação: variável que identifica e diferencia máquinas entre si.

- Condição de Operação: máquinas possuem três modos de operação:

  - normal - a máquina está operando sem restrições;

  - saturada - quando o número de tarefas em fila supera a capacidade do "buffer" de entrada reservado à máquina.

  - pane - a máquina se encontra avariada, e portanto não realiza nenhuma de suas funções.

- Condição de Saturação: quando este valor é igualado ou ultrapassado a máquina se encontra saturada.

- Localização: coordenadas cartesianas que posicionam a máquina no interior do sistema, em relação ao "buffer" celular, arbitrado como origem pelo sistema.

Sistema de Transferência

Entidade responsável pelo transporte das tarefas entre as células, e destas com o armazenamento central.

Atributos:

- Identificação: variável que identifica o sistema de transferência utilizado.

- Velocidade: velocidade média de atuação do sistema de transferência, em unidade compatível.

- Condição de Operação: O sistema de transferência possui dois modos de operação:

  - rotina - o Sistema de Transferência se encontra operando sem qualquer restrição;

  - em pane - o Sist. de Transferência se encontra

avariado, não realizando suas funções de transporte.

De forma análoga ao que ocorre ao manipulador, as rotas possíveis ao Sistema de Transferência são fornecidas em termos das distâncias entre as células e destas com o armazenamento central. O tempo requerido para o transporte é calculado a partir destas distâncias e da velocidade média do equipamento.

Foi assumido que o sistema de controle é capaz de pré-alocar o Sistema de Transferência, não sendo portanto contabilizados atrasos decorrentes de sua movimentação desocupado.

O modelo suporta mais de um equipamento, com a restrição de que todos devem ser do mesmo tipo. No futuro, deverá ser previsto que vários tipos de equipamentos componham o Sistema de Transferência.

Armazenamento Central

Entidade responsável pelo armazenamento das novas tarefas que chegam ao sistema, e pelo armazenamento daquelas que por motivo de pane nos manipuladores ou pane ou saturação em máquinas, encontram-se impedidas de serem recebidas pelas células.

Atributos:

- Localização: coordenadas cartesianas que posicionam esta entidade no interior do sistema, em relação a um referencial arbitrado pelo usuário.

Tarefas

Entidades que representam a produção do sistema. São decompostas em atividades, que ao serem executadas conduzem a um dos produtos finais do sistema.

Estas entidades circulam pelo sistema em "pallets", que são ajustados pelo Sistema de Controle automaticamente às dimensões da tarefa e ao número de peças que compõem o lote. Esta tarefa é executada na entidade carregamento, e portanto foge ao escopo do modelo.

Atributos:

- Identificação: variável que define e diferencia a tarefa.
- Tempo de demanda.
- Atividades: seqüência de operações a serem realizadas sobre as tarefas. é composto de:
  - Primeira Opção de Máquina;
  - Segunda Opção de Máquina;
  - Terceira Opção de Máquina;
  - Tempo de Operação em cada opção de Máquina;
- Tamanho de lote: número de peças que compõem a tarefa.
- Tempo de "Set Up" do lote: tempo necessário para trocar em uma máquina duas peças consecutivas de um mesmo lote.

Prioridades em Fila

Ordenação de retirada das tarefas nas diversas filas que se formam no sistema.

Exemplos de prioridades são:

- FIFO ("first-in-first-out")- primeiro que chega é o primeiro que sai. A ordenação é estabelecida pelo tempo de entrada na fila;
- Prioridade entre tarefas - define-se previamente uma prioridade para cada tarefa, e a ordenação na fila é realizada por este atributo em ordem decrescente. Entre tarefas de mesma

prioridade, FIFO.

- Número de Operações a Completar - será dada prioridade à tarefa que possuir menor número de operações a ser completada. Aquela que estiver mais próximo da saída do sistema, em termos de operações em máquina, será priorizada em atendimento.

- Tempo de entrega Final da Tarefa ("due-date") - o parâmetro de ordenação é o menor tempo de entrega ao cliente de uma tarefa.

Diversas outras prioridades são sugeridas na literatura (GONZALES, 1977), sendo a estrutura do modelo suficientemente flexível para incluí-las com facilidade.

Seqüências Alternativas

Cada operação em máquina poderá ser realizada por três opções de máquina, permitindo uma flexibilização ao modelo que vem ao encontro do conceito de flexibilidade de roteamento apresentado em 2.1.1.

Desta forma, ao término de uma determinada operação serão verificadas as condições de operação (pane, saturação ou normal) das opções especificadas para a próxima operação em máquina. Caso a primeira opção se encontre em operação normal a escolha recairá sobre a mesma. Caso contrário, será selecionada a máquina, dentre as duas opções restantes (em ordem decrescente de prioridade), a que apresenta as condições necessárias para a realização da operação. Se todas as opções não se encontrarem em condições, seja por saturação ou pane, a tarefa será conduzida para o armazenamento central ou para um dos "buffers" celular, dependendo da posição da tarefa, onde deverá ficar em uma condição passiva até que alguma de suas opções de máquina esteja pronta para

recebê-la.

Aleatoriedade dos Processo

Vários eventos que ocorrem ao nível de chão de fábrica são modelados estocasticamente. Assim, foi incluído no modelo um gerador de variáveis aleatórias que servirá de subsídio para a geração dos seguintes eventos:

- Tempo de chegada de uma tarefa no sistema;
- Tamanho do lote que compõe uma tarefa;
- Tempos de operação em máquina;
- Tempos de duração de quebras de máquinas;
- Tempos de duração de reparos de máquinas;

Dentre as distribuições de probabilidades freqüentemente usadas destacam-se:

- Exponencial;
- Normal;
- Uniforme;
- Poisson;
- Hipergeométrica;
- Binomial Negativa;
- m - Erlang;
- Gama;

Todas estas estão disponíveis no sistema.

Fluxo Lógico

A figura 3 apresenta uma descrição de como se processa o fluxo das tarefas no interior do sistema.

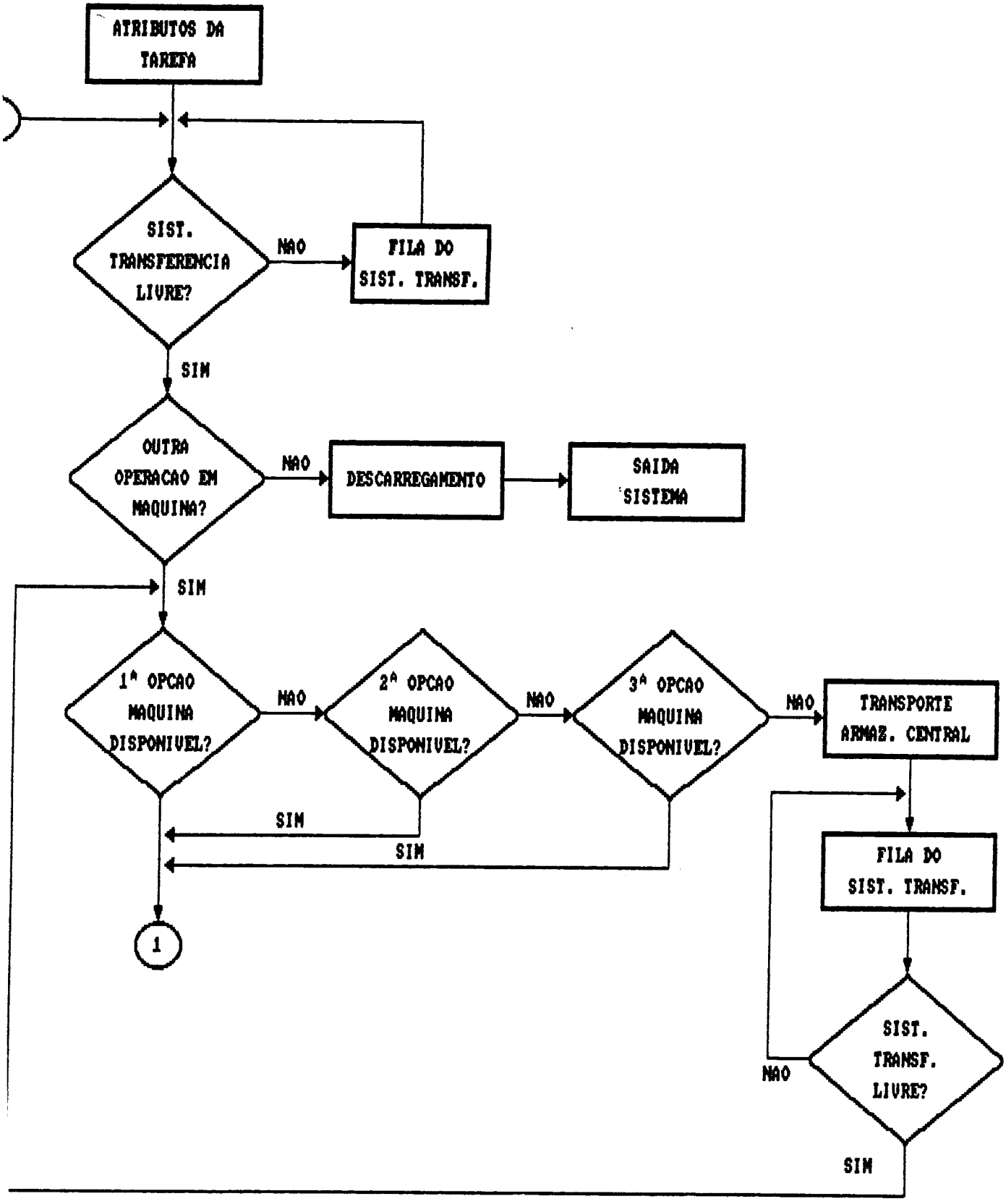


Fig. 3a - Fluxo Lógico do Modelo de Simulação

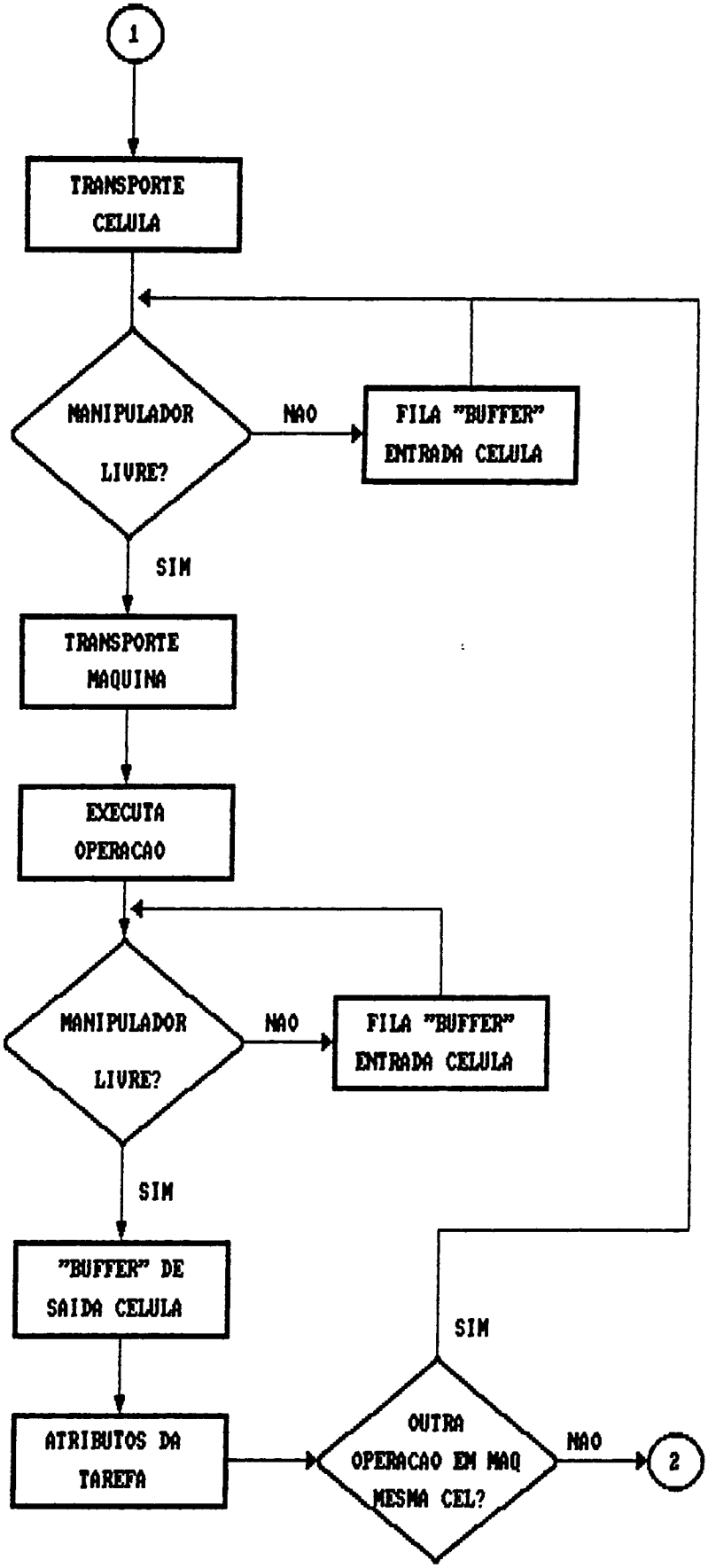


Fig. 3b - Fluxo Lógico do Modelo de Simulação

Quebras de Máquinas e Manipuladores

As quebras de máquinas e manipuladores foram modeladas estocasticamente. A ocorrência de um evento desta natureza acarretará que a entidade envolvida ficará inapta a exercer às funções de sua competência.

Quando da quebra de alguma entidade, o sistema imediatamente se encarregará de redistribuir entre as entidades similares as tarefas que estavam para aquela escalonadas. Caso esta entidade seja insubstituível para uma determinada tarefa, a mesma será deslocada até o armazenamento central ou o "buffer" da célula até o restabelecimento do equipamento avariado.

8. SISTEMA SIMFLEX

O sistema SIMFLEX é a tradução em linguagem computacional do modelo apresentado no capítulo 7. Foi desenvolvido em Turbo Pascal 5.0 (BORLAND, 1988), ocupando um espaço de memória de 140 "Kbytes", sendo compatível com qualquer microcomputador da linha IBM-PC, com memória RAM superior a 512 "Kbytes". Sugere-se sua utilização em micros que possuam disco rígido, a fim de aumentar a eficiência de manipulação. O anexo A apresenta sua listagem.

8.1 - A Escolha da Linguagem

Um dos passos mais importantes no processo de desenvolvimento de um simulador é a definição da natureza da linguagem do programa de simulação, entre as de uso geral (como: BASIC, PASCAL ou C) ou as direcionadas à simulação (como GPSS ou SIMSCRIPT). Esta escolha definirá dois pontos fundamentais em um simulador computacional: eficiência e flexibilidade.

Este assunto é discutido com detalhes na bibliografia sobre simulação (LAW & KELTON, 1982; CARRIE, 1988), que apresenta orientações gerais, baseadas em experiências passadas, de como realizar esta escolha dada uma determinada situação.

Linguagens orientadas à simulação apresentam a vantagem de oferecerem estruturas naturais para simulação, com rotinas capazes de: gerar números randômicos e variáveis aleatórias, avançar o relógio de simulação, passar o controle para outras rotinas, coletar e analisar dados (geração de relatórios). Contudo, elas apresentam uma grande desvantagem em relação a linguagens de uso geral, que é a sua inflexibilidade em termos de acesso lógico da linguagem e de sua estrutura de dados (CARRIE, 1988).

Considerando-se a complexidade do modelo descrito no item anterior, estes fatores foram assumidos como de fundamental importância para o seu desenvolvimento computacional, sendo portanto a escolha direcionada para uma linguagem de uso geral.

Dentre as linguagens disponíveis no PPGA/UFRGS, a escolha final recaiu sobre a linguagem PASCAL, pelas seguintes qualidades:

- uma inerente natureza estruturada;
- sua poderosa capacidade de detecção de erros;
- confiabilidade e portabilidade de seus programas,

bem como pela existência do compilador Turbo Pascal 5.0 (Borland, 1988), que fornece um completo ambiente de programação em microcomputadores, a tornaram ideal a este tipo de aplicação.

8.2 - Estrutura do Sistema

O sistema foi basicamente estruturado em três módulos:

- Módulo Diálogo;
- Módulo Execução;
- Módulo Relatório.

A figura 4 apresenta o inter-relacionamento destes módulos.

Os módulos Diálogo e Execução foram desenvolvidos de forma independentes, de maneira que o usuário poderá ter um acesso individualizado ou a ambos, atravéz do "menu" de abertura do programa, exemplificado na tela 1.

8.2.1 Descrição do Módulo Diálogo

O módulo Diálogo tem como função gerar, em termos amigáveis, um modelo compatível com o módulo Execução do programa SIMFLEX, a partir de um modelo idealizado pelo usuário.

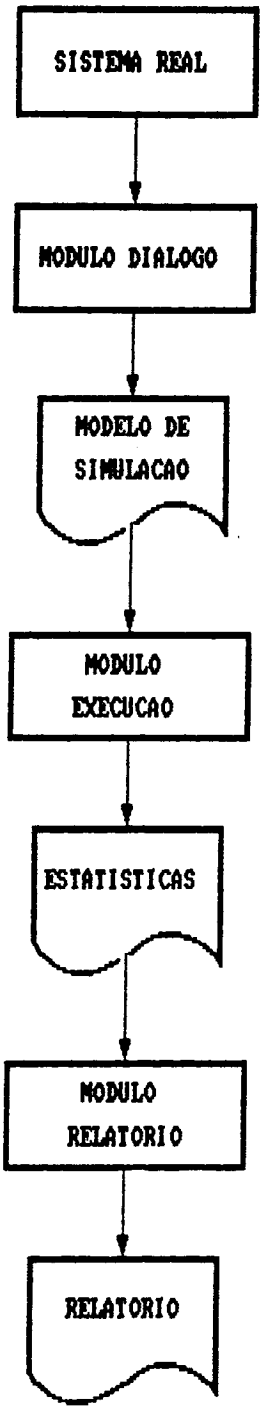


Figura 4 - Fluxo logico do Sistema SIMFLEX

O produto final desta interação usuário/módulo-diálogo é a criação de dois arquivos em ASCII, cujos conteúdos são:

- Definição de todos os atributos das entidades do modelo apresentadas no item anterior, com exceção de suas localizações;
- Definição da localização das entidades do modelo.

Como a definição das localizações das entidades deverá ser

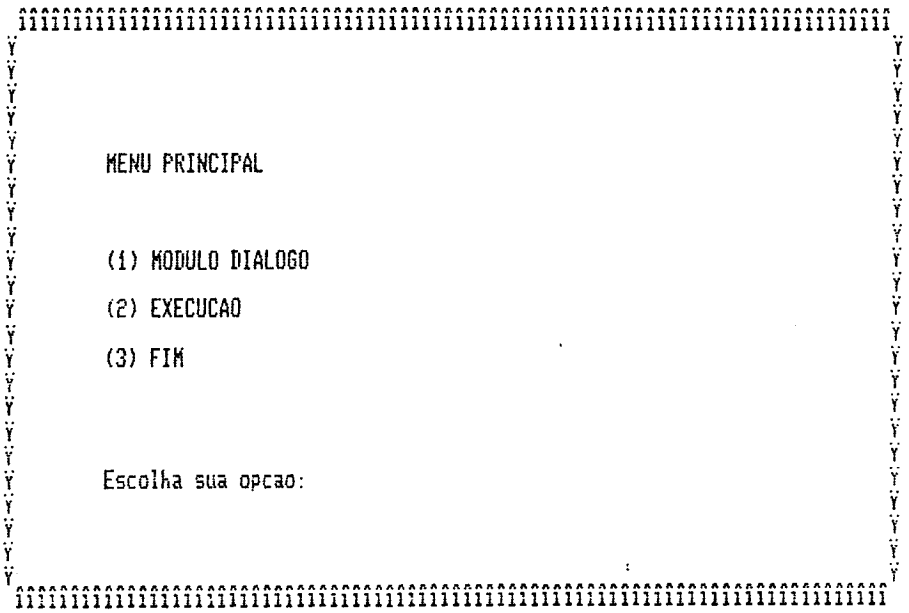


Fig. 5 - Tela 1: Menu de Abertura do Programa

somente obtida, em muitos casos, no final da fase de planejamento, foi decidido pela separação deste atributo dos demais, concedendo ao usuário a opção de definir seu modelo com ou sem o mesmo, evitando deste modo sobrecarga desnecessária na entrada de dados. Neste caso, não serão contabilizadas estatísticas relacionadas com o Sistema de Transferência. Os arquivos criados possuem a característica de serem auto-documentados, onde cada dado de entrada fornecido pelo usuário é precedido de um comentário, indicando a sua função, como apresentado no anexo B. Isto possibilita ao usuário realizar modificações e identificar erros fácil e rapidamente, bastando para tal utilizar qualquer editor de texto, que trabalhe com arquivos em ASCII.

A base da interação é feita através de um conjunto de telas, compostas de 2 a 6 perguntas, organizadas segundo os critérios de

lógica do modelo e de amigabilidade ao usuário. O sistema só progride para a próxima tela, caso todas as informações dadas em resposta às questões da tela atual receberem o aval do usuário. Faz-se a seguir uma demonstração deste módulo.

Inicialmente, o usuário deverá fornecer quatro informações:

- "drive" do arquivo: nome do "drive" (incluindo o "path") em que se deseja gravar o arquivo a ser criado.

Ex.: C:\Simula\FMS

- o nome do arquivo a ser criado;

- o nome da simulação, identificando-a e diferenciando-a de outras já realizadas ou a realizar;

- número de células do sistema. Este número inteiro está limitado, nesta fase de desenvolvimento, a 20.

Um exemplo da tela em que estas informações são requisitadas é apresentado a seguir.

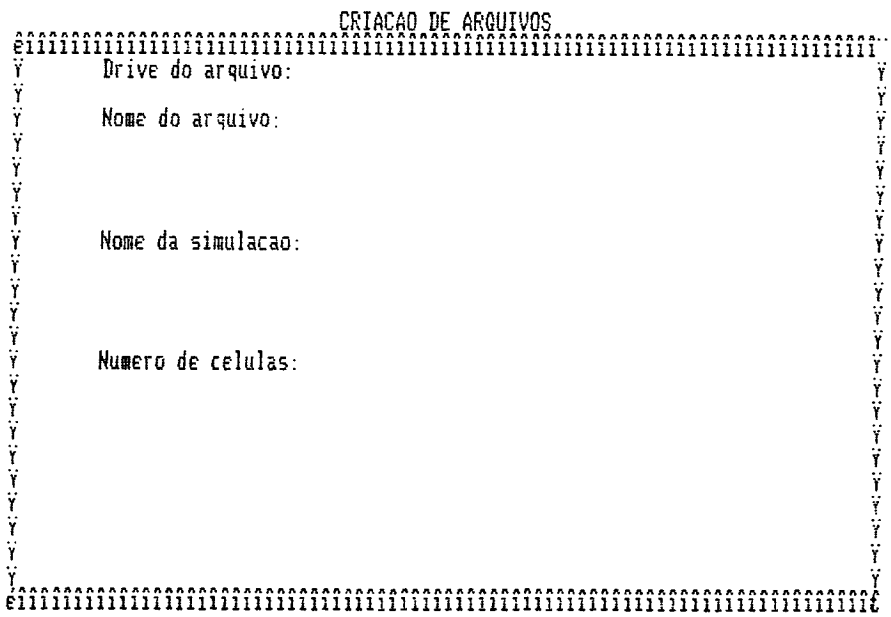


Fig. 6 - Tela 2: Abertura do Módulo Diálogo

A partir do número de células fornecido, serão geradas um

conjunto de telas análogas, que definirão as características de cada célula, sendo a identificação fixada pelo sistema a fim de evitar incompatibilizações futuras.

A primeira tela deste conjunto requisita os seguintes dados:

- Número de Máquinas: número inteiro entre 1 e 5, que informa o número de máquinas no interior da célula i. O valor cinco foi fixado de acordo com as recomendações de GREENWOOD (1989), que o considera como o valor máximo para que não haja rupturas na estrutura celular do FMS.

Para cada máquina definida nesta célula o usuário deverá definir:

- Tempo de "set-up" - valor real que se refere ao tempo necessário para a troca de ferramentas da máquina.

- Nome da máquina: variável alfanumérica (de no máximo 25 caracteres) que identifica o modelo da máquina.

Ex.: Torno Singer T-300

- Identificação da Máquina: número inteiro, variando de 1 a 50, que identifica máquinas de um mesmo modelo, e que definirá a máquina responsável por uma determinada operação em uma tarefa qualquer. Deve-se observar que máquinas de um mesmo tipo no interior de uma célula recebem mesma identificação, o mesmo não ocorrendo caso se encontrem em células distintas.

- Condição de Saturação: valor inteiro positivo que informa o valor máximo em fila , a partir da qual a mesma se encontrará em condição de operação saturada. Caso esta máquina possua capacidade infinita de tarefas em fila, deverá ser fornecido o valor 0 ou teclado diretamente <ENTER>.

A tela 3 apresenta um exemplo de como estas informações são

solicitadas.

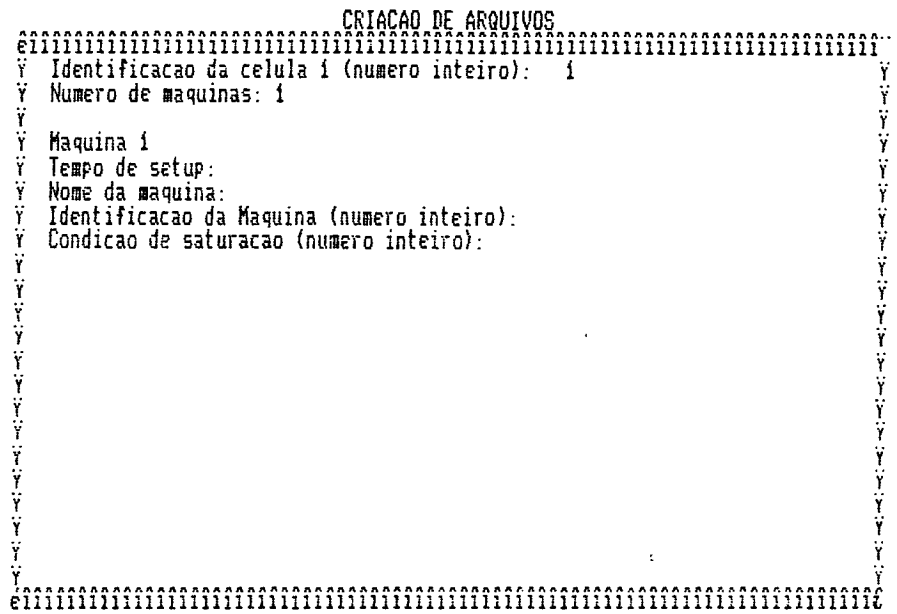


Fig. 7 - Tela 3: Definição da Entidade Célula

A cada máquina existente no sistema estão previstas distribuições de probabilidade que modelam estocasticamente os tempos de ocorrências de avarias e seus respectivos tempos de reparo. Estas distribuições são apresentadas na tela em uma janela que se abre na parte inferior do vídeo, fornecendo o necessário subsídio ao usuário para que forneça a função previamente determinada. A tela 4 exemplifica este processo.

Uma vez determinado o número inteiro que representa a função desejada, será aberta uma nova janela, solicitando os parâmetros que definem matematicamente a mesma. A terminologia empregada é baseada em COOKE et alii(1985) e LAW & KELTON(1982).

Seria por demais extenso uma explanação detalhada de todas as possíveis janelas a serem abertas, já que estão disponíveis dez funções distintas. Recomenda-se ao usuário uma análise destas janelas através de um teste. A título de exemplo, a tela 5

ilustra a janela de parâmetros, para a função normal.

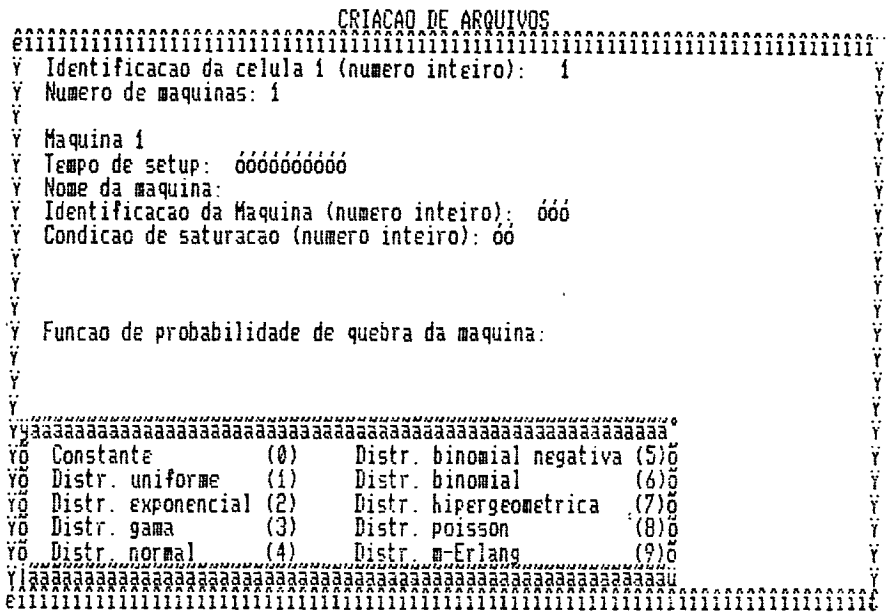


Fig. 8 - Tela 4: Escolha das Funções de Probabilidade

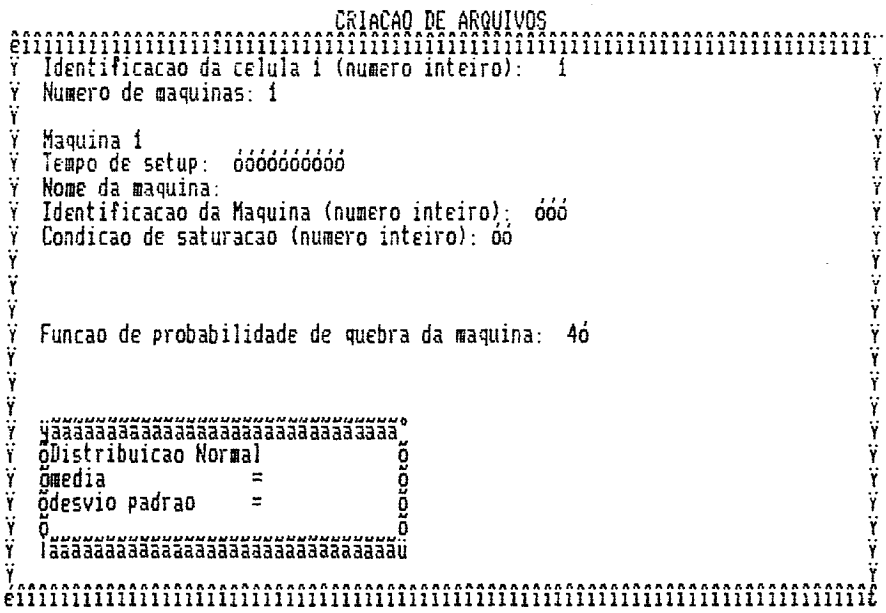


Fig. 9 - Tela 5: Parâmetros da Função Normal

Ao final da identificação de todas as máquinas, será requisitada a identificação do manipulador, da seguinte forma:

- Nome do Manipulador: variável alfanumérica (máximo de 25

caracteres) que identifica o modelo do manipulador.

Ex.:Robô Fanuc S-400

- Velocidade média de atuação do manipulador: valor real.

Fornecida estas informações, encerra-se a definição computacioal de uma célula i, e este processo é repetido para a célula i+1, até que se atinja o número de células fornecido na tela 2.

A próxima tela define junto ao usuário se deseja ou não à inclusão das coordenadas das células, máquinas e demais entidades em seu modelo. Caso fornecido NÃO, o programa será desviado para a definição das tarefas. Caso fornecido SIM, o procedimento abaixo será executado, definindo as coordenadas das entidades.

A tela 6 solicitará ao usuário o "drive" e o nome do arquivo que armazenará às distâncias, de forma bastante análoga à tela 2.

Fig. 10 - Tela 6: Definição do Arquivo de Coordenadas

As próximas telas solicitarão respectivamente as coordenadas da saída do sistema, do armazenamento e "buffers" da célula, em relação a um mesmo referencial, a ser escolhido pelo usuário em seu sistema. Os valores das coordenadas são reais, e dependendo da

origem arbitrada poderão assumir valores positivos e negativos. A tela 7 exemplifica para uma destas entidades.

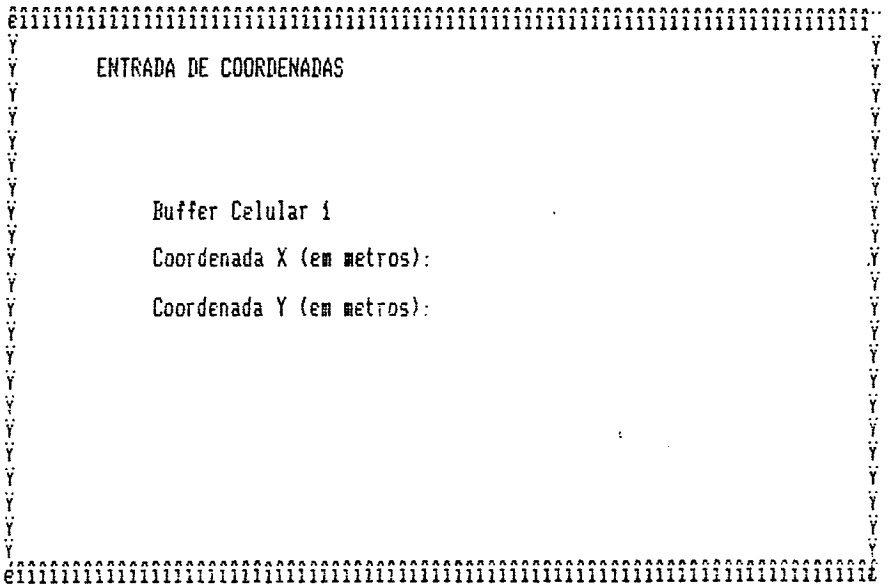


Fig. 11 - Tela 7: Definição das Coordenadas do "Buffer" Celular

A tela 8 exemplifica como são requisitadas as coordenadas das máquinas no interior das células. É importante ressaltar que o sistema arbitra a origem no "buffer" da célula. Os valores são reais, e poderão assumir valores positivos e negativos.

A seguir terá inicio a definição das tarefas, pela solicitação ao usuário da definição do número de tarefas diferentes existentes nesta modelagem, que similarmente ao que ocorre para as células irá definir um conjunto de telas análogas, que definirão os atributos das tarefas.

A primeira tela deste conjunto define:

- Nome do Componente: variável alfanumérica (máximo de 25 caracteres) que identifica a tarefa.

Ex.: Longarina T 20x50x8

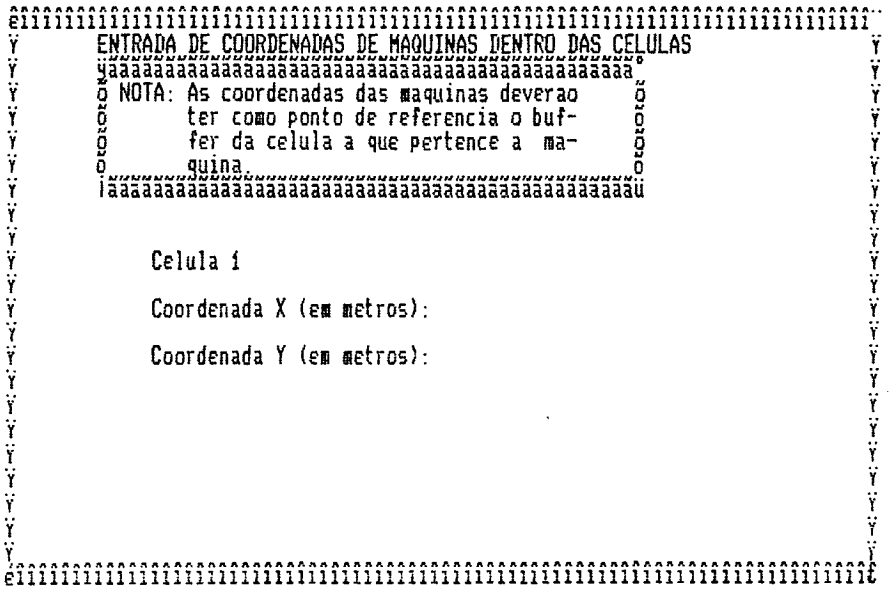


Fig. 12 - Tela 8: Definição das Coordenadas de Máquinas

- Identificação da tarefa: número inteiro, variando de 1 a 100, que diferencia a tarefa das demais.

A tela 9 apresenta como são solicitadas estas informações.

Confirmadas estas informações, o programa solicita a função de probabilidade que rege a geração dos tempos entre chegadas da tarefa no sistema e do tamanho do lote das mesmas. Todo o processo é similar ao descrito para a apresentação das telas 5 e 6.

Neste momento, o usuário definirá o número de atividades em máquina que esta tarefa sofrerá durante sua passagem pelo sistema. A partir deste valor serão definidas um conjunto de telas análogas, que definirão as opções da máquinas de cada seqüência. As mesmas têm como objetivo determinar:

- Identificações das Máquinas de cada uma das três opções disponíveis no sistema: valor inteiro que deverá estar em acordo com os valores fornecidos quando da definição das células. Caso uma tarefa não apresente mais do que uma opção de máquina, ou

somente duas deverá ser fornecido o valor 0 ou <ENTER>, diretamente.

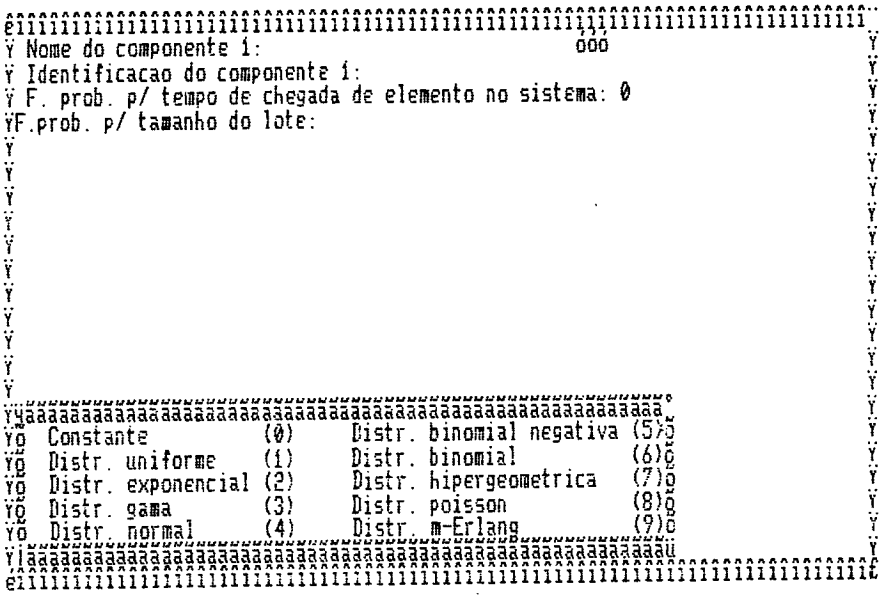


Fig. 13 - Tela 9: Definição da Entidade Tarefa

- Funções de probabilidade que representam estocasticamente o tempo de ocupação da máquina identificada para cada opção. A definição das mesmas é realizada de forma similar a discutida anteriormente para as telas 4 e 5.

A tela 10 exemplifica como ocorrem estas entradas.

Ao final deste conjunto de telas, que definem o conjunto de atributos de uma atividade i, deverá ser definida um nova tarefa i+1, repetindo-se o processo que tem seu início na tela 9. Quando o número de tarefas modelado pelo usuário for atingido, o módulo desviará sua atenção para a definição do Sistema de Transferência, que é composta de:

- Nome do Sistema de Transferência: variável alfanumérica (máximo de 25 caracteres) que identifica o modelo de Sistema de

Transferência a ser utilizado.

Ex.: AGV Eaton-Kenway

- Velocidade média de atuação do Sistema de Transferência: valor real.

- Número de Equipamentos do Sistema de Transferência: valor inteiro de 1 a 5, definindo o número de equipamentos do modelo acima identificado que compõe o sistema idealizado.

Um exemplo da tela de definição do Sistema de Transferência é apresentado na tela 11.

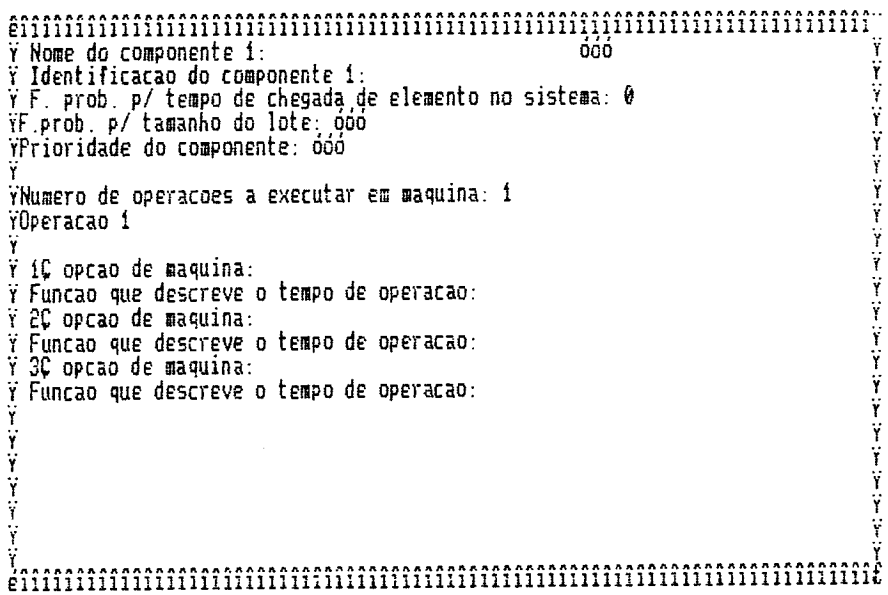


Fig. 14 - Tela 10: Opções de Operação em Máquina

6.2.3 Descrição do Módulo Execução

O módulo execução tem como função executar a simulação do sistema idealizado pelo usuário, e definido com o auxílio do módulo diálogo, fornecendo as suas estatísticas, que serão compiladas pelo módulo relatório, fornecendo parâmetros

fundamentais para uma análise do mesmo.

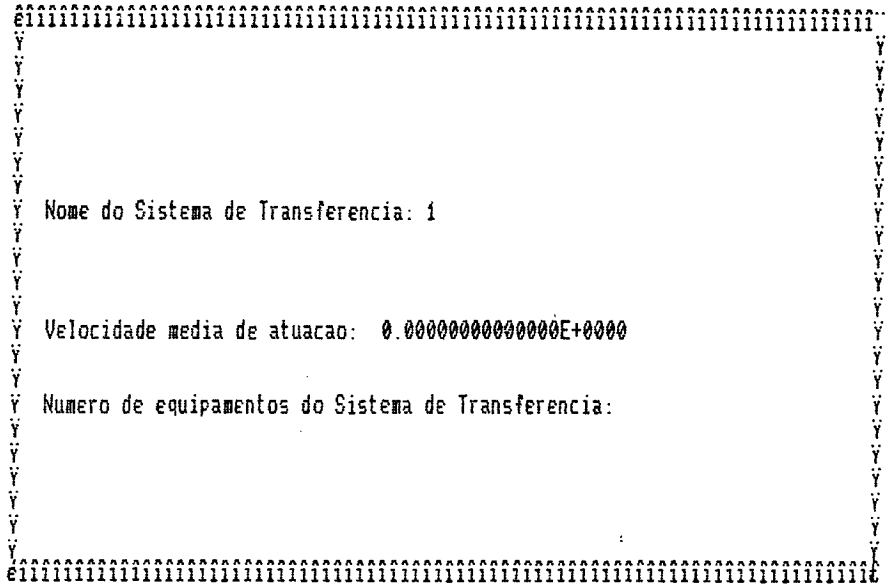


Fig. 15 - Tela 11: Definição da Entidade Sistema de Transferência

O módulo execução foi desenvolvido segundo a abordagem baseada no próximo-evento (LAW & KELTON, 1982). Neste tipo de abordagem uma rotina, denominada de rotina de evento, é escrita para definir cada um dos eventos que ocorrem no sistema. Estas rotinas são responsáveis pela descrição detalhada das mudanças que ocorrem nas variáveis de estado do sistema, quando um determinado evento ocorre. O avanço do tempo corrente de simulação , definido como relógio de simulação, ocorre sempre ao evento mais iminente.

O fluxo lógico de como este módulo foi desenvolvido é apresentado na figura 16.

O módulo Execução apresenta atualmente as seguintes rotinas de evento:

- Saida_Armazenamento: responsável pela saída do

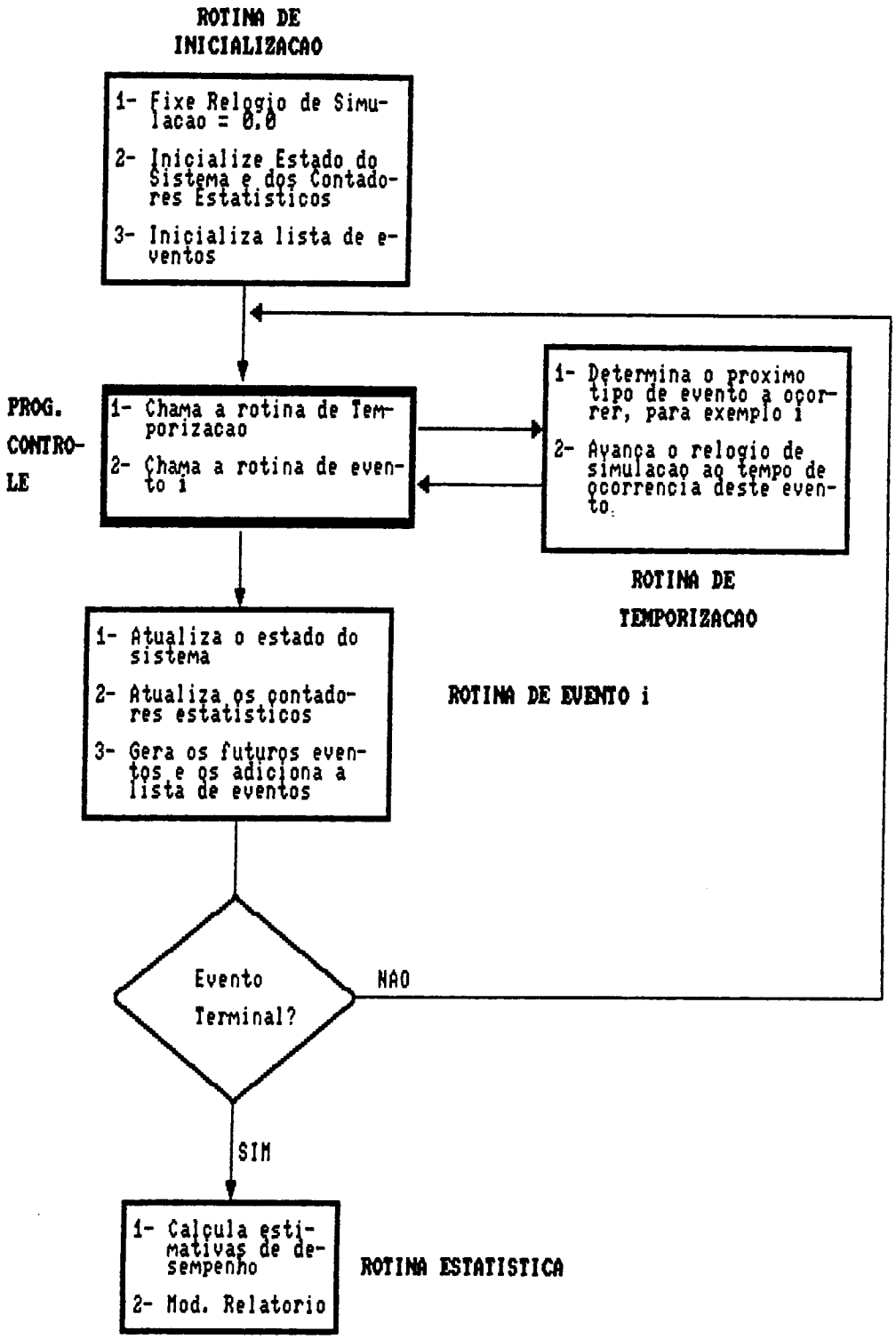


Fig. 16 - Fluxo Lógico do Módulo Execução

armazenamento de uma tarefa para as atividades de manufatura, e conseqüente ocupação do Sistema de Transferência.

- Saída_ST: responsável pela desocupação do Sistema de Transferência por uma tarefa, definindo o lugar no sistema que a mesma ocupará: "buffer" celular, armazenamento ou saída. Caso exista alguma tarefa em fila pronta para ocupar o Sist. de Transferência, a rotina gerará sua ocupação.
- Saída_Manipulador: responsável pela desocupação do manipulador de uma célula i, definindo para onde deverá ser direcionada a tarefa correspondente. Caso exista alguma tarefa em fila pronta para ocupar o manipulador, a rotina se encarregará de fazê-lo.
- Saída_Maquina: responsável pela desocupação de uma máquina, e a posterior ocupação do manipulador celular.
- Quebra_Máquina: responsável pela quebra de uma máquina, retirando-a de operação pelo tempo necessário para a realização de seu reparo, bem como pelo redirecionamento das tarefas em fila desta máquina para as demais opções.
- Reparo_Máquina: responsável pelo retorno de uma máquina, em reparo, para operação, assim como pela geração de uma nova quebra.

Para que o módulo Execução seja realizado, o usuário deverá definir:

- Sistema de Prioridade a ser utilizado nesta corrida;
- Tempo de simulação desejado;

- "Drive" e nome dos arquivos de definição das entidades;

- "Drive" e nome do arquivo que armazenará as estatísticas;

Estas informações serão solicitadas por intermédio de telas auto-explicativas e similares as do módulo Diálogo, dispensando exemplificações.

8.2.3 Descrição do Módulo Relatório

O módulo relatório tem como função criar um arquivo em ASCII, que contém o sistema idealizado pelo usuário, devidamente formatado, e os resultados obtidos pelo Módulo Execução.

A importância de um relatório conciso, preciso e de fácil percepção em um estudo de simulação, pela quantidade de dados gerados, é inegável, dispensando comentários. Porém a sua execução é extremamente trabalhosa, principalmente se forem agregadas facilidades gráficas (gráficos de barras, de torta, animações, etc.). Levando-se em conta a natureza acadêmica deste trabalho, foi dada prioridade ao desenvolvimento dos dois módulos acima apresentados, pois estes solicitam grande esforço de modelagem, em detrimento do módulo relatório, cujo desenvolvimento se baseia, fundamentalmente, em programação.

Desta forma, o relatório atualmente gerado pelo sistema SIMFLEX é bastante simples, fornecendo os resultados sob a forma de tabelas numéricas.

O resultado mais importante a uma análise de um FMS é a taxa de utilização de cada estação de trabalho (ou máquina), pois um alto valor para a mesma é condição fundamental para a viabilidade econômica destes sistemas de manufatura (WARNECK & GERECKE, 1977). Entre outros resultados fornecidos pelo programa,

destacam-se:

- O tempo de espera das tarefas, quando em estado passivo (em fila ou alojada nos "buffers" do sistema);
- A taxa de tarefas produzidas pelo sistema, no tempo de simulação;
- Taxas de utilização do Sistema de Transferência e manipuladores;
- Número médio de tarefas em fila para cada uma das entidades permanentes no sistema.

O objetivo principal destes valores é fornecer ao usuário dados suficientes para que possa comparar sistemas alternativos, permitindo-o escolher aqueles com boa utilização, segundo critérios previamente estabelecidos, de todos os componentes do sistema.

Um exemplo de relatório emitido pelo programa é apresentado no anexo C.

É importante salientar que devido a natureza aleatória dos dados de entrada fornecidos ao simulador, os resultados obtidos em uma corrida de simulação também apresentarão uma natureza aletória, e portanto os resultados de uma corrida isolada podem diferir bastante do real comportamento do sistema em estudo. Como decorrência a este fato, é necessário uma análise estatística dos resultados, normalmente custosa tanto em formulação como em solução computacional. LAW & KELTON (1982) apresentam uma discussão detalhada de como esta análise pode ser realizada. Este tópico foge ao escopo deste trabalho, porém deverá receber atenção especial, após a consolidação do modelo, pela inclusão de um módulo estatístico de análise ao SIMFLEX.

Após a total validação do modelo computacional, que permitirá sua utilização comercial, este módulo deverá sofrer modificações sensíveis, com a inclusão de várias rotinas gráficas, com o objetivo de tornar o simulador mais amigável ao usuário.

8.3.1 Verificação

A verificação de um modelo de simulação computacional consiste no processo de "debbuging" do programa, ou seja, a determinação se o mesmo executa o que dele se deseja (LAW & KELTON, 1982).

Embora este conceito seja simples e evidente, a realização da verificação de um modelo de simulação computacional de grande porte é extremamente trabalhoso, exigindo grande esforço computacional e de programação.

Para a verificação do sistema SIMFLEX foi utilizado um "mix" de três técnicas bastante difundidas (SCHMITZ & TELES, 1986):

Técnica 1 - O programa foi dividido em módulos, sub-rotinas, que eram testadas logo após sua implementação computacional.

Técnica 2 - Realizando-se o "trace" do programa, isto é observando o seu funcionamento passo a passo.

Técnica 3 - A inclusão no programa de rotinas de escrita, permitindo a fotografia do sistema a cada evento, pela impressão das variáveis de estado.

A técnica 1 foi utilizada para as rotinas mais básicas do programa, como manipulação dos dados, geração de variáveis aleatórias, etc. Já as duas técnicas restantes foram utilizadas

para as rotinas mais complexas, como as rotinas de evento. Para a aplicação da segunda técnica, foi fundamental a contribuição do compilador Turbo Pascal.

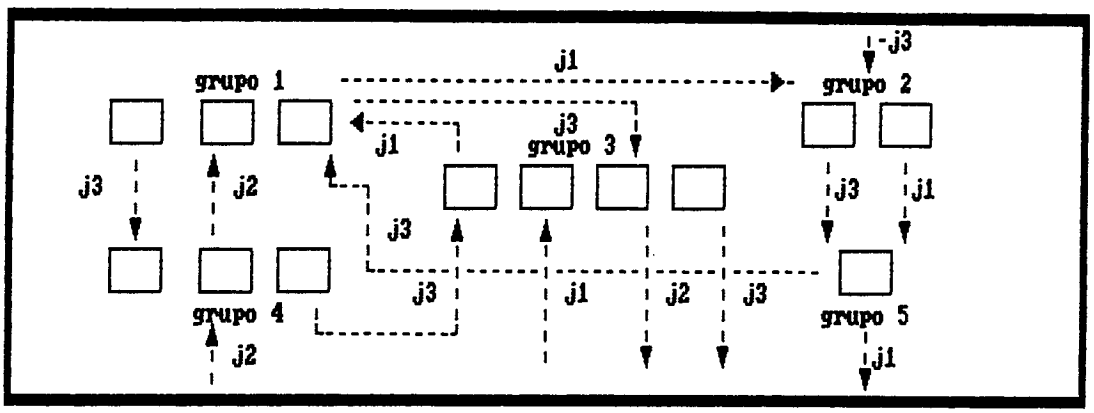
Para que o processo de verificação fosse realizado de forma racional, o modelo foi desenvolvido tendo-se como maior preocupação a sua capacidade evolutiva, ou seja, a capacidade de agregar complexidades por degraus.

O primeiro grande teste foi realizado quando da finalização da implementação da entidade célula, que pode ser considerada o elemento básico do modelo genérico. Para tanto foi utilizado o Modelo JOB-SHOP apresentado por LAW & KELTON (1982), e traduzido em linguagem computacional pelo programa SIMLIB. Um sumário do modelo pode ser observado na figura 17.

Os resultados apresentados pelo SIMLIB e pelo SIMFLEX, simplificado, são apresentados na tabela 2, considerando-se um tempo de simulação igual a 2920 h.

Pela tabela abaixo apresentada pode-se observar que os resultados apresentam uma boa conformidade, tendo-se em vista a aleatoriedade dos processos. Este fato permite afirmar que o modelo, até aquela fase, encontrava-se verificado, bem como pode ser rapidamente adaptado para representar um JOB-SHOP convencional, comprovando uma das maiores preocupações durante o desenvolvimento do modelo: a sua flexibilidade.

MODELO JOB-SHOP



5 tipos de máquinas

3 tipos de "jobs" (Produtos)

Processamento em ordem pré-especificada

Processo de chegada: IID - exponencial com médias entre chegadas de 0.25

Distribuição por tipo de produto: 1 - 30%; 2 - 50%; 3 - 20%

Processo de Serviço: IID - 2-Erlang com médias iguais a (em h)

Produto	Máquinas na Seqüência
1	3:0.5 ; 1:0.6 ; 2: 0.85 ; 5:0.5
2	4:1.1 ; 1:0.8 ; 3:0.75
3	2:1.2 ; 5:0.25 ; 1:0.7 ; 4:0.9 ; 3:1.0

Fig. 17 - Descrição do Modelo JOB-SHOP de LAW & KELTON (1982)

Ident. Comp.	Tempo Medio Espera	
	SIMLIB	SIMFLEX
1	13.293	12.579
2	8.066	10.685
3	17.879	19.065

Tab. 2a - Comparação resultados SIMLIB - SIMFLEX

Maq.	No. médio Fila		Perc. Ocupação	
	SIMLIB	SIMFLEX	SIMLIB	SIMFLEX
1	11.192	15.145	0.950	0.952
2	17.999	15.017	0.958	0.978
3	0.725	0.750	0.721	0.718
4	14.084	18.405	0.966	0.959
5	1.926	1.624	0.796	0.778

Tab. 2b - Comparação resultados SIMLIB - SIMFLEX

Existe uma relação direta entre o sistema FMS e a entidade célula no modelo abstrato construído. Ou seja, a célula pode ser vista como o FMS, pala equivalência funcional existente entre as entidades manipulador-Sistema de Transferência e máquina-célula. Isto torna a entidade célula a unidade básica do modelo. Esta hierarquização proposital teve como objetivo facilitar tanto a programação como a verificação do SIMFLEX.

Tendo-se a estrutura básica célula totalmente testada, a verificação do sistema SIMFLEX pôde ser realizada de forma simplificada, pelo teste individualizado das rotinas incrementadas a esta estrutura. O inter-relacionamento das mesmas

foi garantido via a criação de modelos fictícios com nível de complexidade compatível com as rotinas a serem testadas.

9. CONSIDERAÇÕES FINAIS

Neste capítulo são apresentadas considerações finais sobre o processo de modelagem realizado, fazendo-se uma avaliação crítica do Sistema SIMFLEX. Recomendações para a realização de trabalhos futuros que venham a expandir o atual também são apresentadas.

Os benefícios apregoados pelo conceito Sistema Flexível de Manufatura, como redução do "lead-time", redução de "work-in-process" e aumento de flexibilidade (principalmente em termos de "mix" de produtos), não estão sendo verificados quando de sua efetiva aplicação prática (TCHIJOV & SHEININ, 1989). A principal causa para este fenômeno é a barreira existente entre a concepção teórica e a sua aplicação prática, justificada face a alta tecnologia, complexa definição e elevado custo de capital inerentes a este conceito. Para que esta barreira seja ultrapassada é necessário o desenvolvimento de ferramentas objetivas de análise capazes de captarem os fenômenos que ocorrem no interior destes sistemas, fornecendo parâmetros de desempenho sobre as mais variadas condições de operação.

Modelos de simulação, pela capacidade de agregar eventos estocásticos e pela forma dinâmica como os manipulam, apresentam-se como representações ideais, senão únicas, para esses sistemas de manufatura. Contudo, a falta de profissionais com conhecimentos simultâneos em simulação e FMS tem se apresentado como uma dificuldade difícil de ser transposta. A consciência deste fato pelos especialistas em Pesquisa Operacional, acarretou em uma tendência de se desenvolverem modelos genéricos de simulação. Estes modelos se apresentam de uma forma tal que os parâmetros do sistema a ser modelado podem ser alterados pelo

usuário através dos dados de entrada, sem a necessidade de modificações nas rotinas que compõem o modelo. Isto evita completamente qualquer programação por parte do usuário, permitindo que o mesmo só se preocupe com a modelagem do sistema a ser analisado.

Um grande número de sistemas simuladores desta natureza já se encontram disponíveis comercialmente em países que utilizam o conceito de FMS. Estes simuladores geralmente são desenvolvidos por especialistas que estudaram e trabalharam em uma grande variedade de projetos de FMS, e que incluem em seus modelos o maior número possível de características especiais exibidas por estes sistemas. Caso o sistema a ser modelado é uma configuração padrão similar a outras já existentes, estes pacotes apresentam um excelente desempenho (CARRIE, 1988). Porém, caso o mesmo envolva caraterísticas especiais, como uma inovação tecnológica, o mesmo já não ocorre. A principal fraqueza destes modelos genéricos é a freqüente incapacidade de manipular sistemas com atributos especiais, que atualmente são a grande maioria dos FMS em projeto ou instalação (GREENWOOD, 1989). Estes problemas são decorrentes do segredo imposto pelos fabricantes da estrutura lógica do modelo gerado pelos mesmos, a partir das informações do usuário, e quer acabam produzindo modelos impossíveis de serem validados.

A partir dessas constatações, pode-se concluir que o fator de sucesso de um modelo genérico é a sua capacidade em fazer o usuário entender sua estrutura básica, ou seja, o modelo abstrato a partir do qual todos os modelos de aplicação serão gerados. Pois assim, estará se oferecendo uma ferramenta versátil, na qual

alterações poderão ser realizadas sem a necesssidade do especialista que idealizou o pacote. O que se observa é que realizar estas atividades em um pacote comercial é quase impossível, pois as suas estruturas básicas se escondem em sigilo industrial e em linguagens de difícil compreensão ou de rara documentação. Isto é agravado em países como o Brasil, onde não existe tradição na utilização de simulaçao como técnica de análise do desempenho de sistemas avançados de manufatura.

É neste contexto que surge a motivação de desenvolvimento do Sistema SIMFLEX, um modelo genérico que seja capaz de sobrepujar estas deficiências acima apresentadas, diferenciando-se dos já existentes, oferecendo um modelo genérico flexível por excelência. A obtenção deste atributo foi conseguida pela priorização ao desenvolvimento de um sistema em que a concepção teórica superasse às capacidades computacionais, utilizando-se a estratégia de não se preocupar em tentar representar todas as particularidades existentes em sistemas descritos na literatura, mas sim em desenvolver um modelo abstrato hierárquico simples, independente de particularidades de determinado sistema, e portanto capaz de representar, com alterações pertinentes, qualquer tipo de FMS. Pequenas alterações são suficientes para tornar o sistema SIMFLEX apto para representar qualquer sistema que futuramente venha a ser implementado no Brasil. O principal sub-produto deste esforço é o conhecimento sobre o conceito FMS adquirido, fazendo com que o SIMFLEX não se torne um mero similar nacional a pacotes importados.

Desta forma, acredita-se que o esforço gasto na idealização e construção do Sistema SIMFLEX é plenamente justificado, podendo

ser considerada uma ferramenta fundamental dentro do processo de modernização de nosso parque industrial.

9.1 Avaliação Crítica do Sistema SIMFLEX

O sistema SIMFLEX encontra-se totalmente desenvolvido e em funcionamento. Deve ser considerado um protótipo, na medida em que se encontra em processo de validação, dificultado pela ausência de sistemas de manufatura no Brasil orientados pelos conceitos de FMS (vide capítulos 2 e 3). Até o momento, o sistema foi testado utilizando-se exemplos descritos na literatura (LAW & KELTON, 1982) e de modelos idealizados no âmbito do PPGA/UFRGS.

Embora o baixo período de experimentação com o sistema simulador, torna-se aparente que o mesmo corresponde aos objetivos especificados no capítulo 5.

A capacidade do sistema de aceitar diversos "lay-outs" e "mix" de produtos distintos, aliado a um número significativo de algoritmos responsáveis pelo controle celular e central (ordenação das tarefas em fila, esquemas que limitam a movimentação de tarefas por questões de quebra ou saturação de equipamentos), dotam o sistema SIMFLEX de grande flexibilidade e generalidade, requisitos fundamentais para um sistema simulador com as suas características. Estes atributos foram obtidos, principalmente, pelos seguintes aspectos:

- A estrutura do modelo abstrato. A estrutura hierárquica definida, com uma analogia funcional entre as entidades que compõem a célula e o sistema FMS (o FMS pode ser visto como uma célula, em que o manipulador equivale ao Sistema de Transferência e as máquinas às celulas, e vice-versa)

pode ser considerado o fator de êxito desta pesquisa. A estratégia de se priorizar a definição do modelo abstrato em detrimento de sua programação imediata também pode ser considerada importante para este êxito.

- A estrutura de dados e a estrutura lógica da linguagem utilizada.A poderosa estrutura de dados e lógica do Turbo Pascal 5.0 concedeu grande flexibilidade de programação das rotinas idealizadas, bem como alta rapidez em suas verificações. Pode-se considerar este ambiente de programação como ideal para a programação de modelos de simulação complexos e genéricos.

Como qualquer protótipo o sistema desenvolvido apresenta limitações, originadas tanto das limitações da estrutura modelar definida, como por problemas relacionados com a programação e utilização da memória originadas de deficiências do autor.

As limitações decorrentes da estrututra montada para o modelo são:

- As células só podem ser definidas com um e somente um manipulador. Este fato corresponde à realidade de sistemas totalmente automatizados (GREENWOOD, 1989). Contudo o mesmo não ocorre para sistemas semi-automatizados, onde existem vários manipuladores no interior de uma célula. Isto acarretará dificuldades quando da modelagem a ser realizada pelo usuário para a utilização efetiva do SIMFLEX.
- O sistema de transferência e os manipuladores só podem transportar um "pallet" por vez. Isto significa que esteiras rolantes e equipamentos similares, capazes de

transportar mais de um "pallet" simultaneamente, não receberão representação adequada. Para algumas situações este fato poderá ser solucionado adequando-se o número de equipamentos do sistema de transferência a capacidade do equipamento real. Para os manipuladores, infelizmente, esta sugestão não poderá ser empregada.

Para que estas limitações sejam superadas será necessário a inclusão de um algoritmo de controle cuja função seja a de selecionar rotas que minimizem o tempo de transporte realizado por esses equipamentos, dada uma determinada condição.

Com relação ao sistema SIMFLEX, visto como um programa, as seguintes limitações abaixo merecem destaque.

- Tempo de execução. O tempo de execução para uma determinada corrida de simulação poderá ser bastante grande, decorrência direta da generalidade propositadamente concedida ao sistema.
- Espaço de Memória.O número de células e de tarefas foram limitados a 20 e 100 respectivamente, a fim de garantir espaço de memória suficiente para a execução da simulação.

Estas limitações estão diretamenta associadas às limitações impostas pelo sistema computacional em utilização, microcomputadores da linha IBM-PC. Com o crescente avanço tecnológico destes equipamentos, acredita-se que as mesmas terão seus efeitos minimizados substancialmente.

9.2 Potencial de Utilização do Sistema

O grande mérito do sistema SIMFLEX é a de fornecer

estimativas de desempenho de sistemas avançados de manufatura em laboratório, evitando experimentações com protótipos, extremamente complexas e caras. Seus resultados ao serem analisados sobre critérios previamente estabelecidos permitirão aos usuários tomarem decisões baseadas em múltiplos atributos, alguns de caráter estratégico e organizacional, e de difícil mensuração sem o auxílio de uma ferramenta de análise. Neste sentido, pode-se afirmar que o SIMFLEX se encaixa como elemento fundamental dentro de um processo de tomada de decisão de FMS, fornecendo um instrumento capaz de acessar aspectos intangíveis (qualidade, flexibilidade, redução de "lead time"), vitais a estes sistemas, evitando decisões baseadas somente em fatores técnicos e em custos contábeis de fácil mensuração. Ressalte-se que o SIMFLEX, neste sentido, dará suporte como ferramenta básica ao sistema de apoio à decisão, atualmente em desenvolvimento no âmbito do PPGA/UFRGS.

Outra contribuição do sistema é a sua capacidade de familiarizar gerentes e administradores com o nível operacional de um sistema de manufatura, facilitando a percepção das novas filosofias existentes na área de produção; constituindo-se assim em excelente instrumento de treinamento e instrução de pessoal relacionado direta e indiretamente ao processo de tomada de decisão relacionado a FMS.

9.3 Recomendações

As seguintes recomendações podem ser sugeridas no sentido de expandir este trabalho:

- A validação do SIMFLEX em um sistema real, definindo se os

seus resultados representam com acurácia os processos que ocorrem nos mesmos.

- Ampliação do escopo do modelo abstrato, pela inclusão de aspectos logísticos, como: manipulação da matéria-prima, arrumação das tarefas nos "pallets", entrada de tarefas no sistema segundo previsões de vendas, etc.
- Inclusão no sistema SIMFLEX de rotinas responsáveis por:
  a - Impressão gráfica dos resultados, facilitando o entendimento dos mesmos pelos usuários.
  b - Análise estatística dos resultados, poderosa o suficiente para garantir confiabilidade aos mesmos durante o processo de tomada de decisão. Os métodos estatísticos capazes de realizar esta análise podem ser encontrados em LAW & KELTON (1982).
  c - Animação do processo de simulação, facilitando ao usuário a compreensão da seqüência de eventos que ocorrem no tempo de simulação, aumentando as potencialidades didáticas deste sistema.

10. REFERÊNCIAS BIBLIOGRÁFICAS

BORLAND. Turbo Pascal - Reference guide, Version 5.0. USA, Borland International, Inc., 1988.

CARRIE, N. Simulation of Manufacturing Systems. USA, John Wiley & Sons, 1988.

CARTER, Charles F. The Role of the Robots in automation Work. In: Nof, Shimon Y. (coord.). Handbook of Industrial Robotics. USA, John Wiley & Sons, p. 9-20, 1985.

CERVO, A. L. & BERVIAN. Metodologia Científica. 2. Ed., São Paulo, McGraw-Hill do Brasil, 1978.

COLLINS, J. A Numerical Control and Flexible Manufacturing Systems. In: INFOTECH Invited Papers, Series 8, U.K., n.6 p. 59-87, 1980.

COOKE, D.; CRAVEN, A. H. & CLARKE, M. G. Statistical Computing in Pascal. U.K., Edward Arnold, 1985.

CORIAT, B. Automação Programável : Novas formas e Conceitos de Organização da Produção. In: Fleury & Fischer (coord.).Processo e Relações de Trabalho no Brasil. São Paulo, Atlas, p. 13-61, 1985.

GONZALES, Mário J. Deterministic Processor Scheduling. Computing Surveys, V. 9, n. 3, p. 174-204, 1977.

GREENWOOD, N. Implementing Flexible Manufacturing Systems. U.K., John Wiley & Sons, 1989.

HASEGAWA, Y. Evaluation and Economic Justification. In: Nof, Shimon Y. (Ed.). Handbook of Industrial Robotics. USA, John Wiley & Sons, p. 665-687, 1985.

HUGHES, J.J. Simulation: Computer-aided Problem Solving. Proceedings of Multi-Station, Digitally Controlled

Manufacturing Systems. University of Wiscosin, 1977.

HUTCHINSON, G. K. The Control of Flexible Manufacturing Systems: Required Information and Algoritm Structures. In: Oshima, Y. (Ed.). Proceedings of the IFAC International Symposium. Tokyo, Japão, Oct. 1977.

INFOTECH State of the Art Report. Factory Automation Analysis. Series 8, U.K., n. 6, 1980.

KOREM, Y. Computer Control of Manufacturing Systemas. USA, McGraw-Hill Book, USA, 1983.

LAW, A. & KELTON, D. Simulation Modeling and Analysis. USA, McGraw-Hill Book, 1982.

MEREDITH, J. & HILL, M. Justifying New Manufacturing Systems: A Managerial Approach. Sloan Management Review. Summer 1987, p. 49-61.

RATHMILL, K. Computer Simulation of the Factory of the Future. In: INFOTECH Invited Papers, Series 8, U.K., n. 6, p. 177-205, 1980.

SARGENT, R. G. Event Graph Modelling for Simulation with an Application to Flexible Manufacturing Systems. Management Science, v. 34, n.10 , p.1231-1251, 1988.

SALERNO, M. Automação e Processos de Trabalho na Indústria de Transformação. XI Encontro Anual da ANPOCS, Águas de São Pedro, 1987.

SCHMITZ, E. A. & TELES, A. A. Pascal e Técnicas de Programação. 2. ed., Rio de Janeiro, LTC - Livros Técnicos e Científicos Editora S.A., 1986.

STRACK, Jair. GPSS: Modelagem e Simulação de Sistemas. Rio de Janeiro, LTC - Livros Técnicos e Cientificos

Editora S.A., 1984.

SURI, Rajan. Quantitative Techniques for Robotics Systems Analysis. In: Nof, Shimon Y. (coord.). Handbook of Industrial Robotics. USA, John Wiley & Sons, p. 605-638, 1985.

TCHIJOV, I. & SHEININ, R. Flexible Manufacturing Systems(FMS): Current Diffusion and Main Advantages. Technological Forecasting and Social Changes, v. 35, p. 277-293, 1989.

WARNECKE, H-J. & GERICKE, E. Modelling and Simulation of Automated Manufacturing Processes. In: Oshima, Y. (Ed.). Proceedings of the IFAC International Symposium. Tokyo, Japão, Oct. 1977.

YOSHKAWA, H. State of the Art in Japan Flexible Manufacturing System. In: INFOTECH Invited Papers, Series 8, U.K., n. 6, p.177-205, 1980.

YTO, Y. Present Status and Trends in Flexible Manufacturing System. Journal of Japan Society of Mechanical Engineers, v. 85, n. 761, 1982.

Listagem das Rotinas que Compõem o Sistema SIMFLEX

```
{$O+,F+}
PROGRAM SIMFLEX;
Uses Overlay,Crt,Dos,tela,menu,abert,Letecla,dialogo,exec;
{$O abert}
{$O menu}
{$O dialogo}
{$O tela}
{$O defenti}
{$O defent2}
{$O coord}
{$O ler}
{$O learq}
{$O dist}
{$O Funcoes2}
{$O Le_funcoes}
{$O Defdist}
{$O Relatori}
{$O Relatres}
var
   resposta:byte;
   c1,c2:byte;
begin
     OvrInit('Simflex.ovr');
     OvrInitEms;
     randomize;
     abertura;
     clrscr;
     menu;
end.

Unit Defvar;
Interface
     Type
     {Definicao de variaveis}
     palavra=string[16];
     status_maq= (ocupado,desocupado);
     cond_operacao_maq=(normal,saturado,pane);
     status_manip= (ocup, nao_ocupado);
     cond_operacao_manip=(rotina,em_pane);
     task=record
               maquina:array [1..3] of byte;
               distribuicao:array [1..3] of byte;
               par_1:array [1..3] of real;
               par_2:array [1..3] of real;
               par_3:array [1..3] of real;
               end;
     ptr_job=^job;
     job=record
               componente:palavra;
               identificacao:byte;
               elemento:array[1..10]of task;
               distr_cheg_armaz:byte;
               par_chegi:real;
               par_cheg2:real;
               par_cheg3:real;
               distr_lote:byte;
               par_l1:real;
               par_l2:real;
               par_l3:real;
               last:byte;
               ptr_proximo:ptr_job;
               end;
   ponteiro=^tarefa;
   tarefa=record
                identificacao:byte;
                rota:byte;
                operador:byte;
                T_ocupacao:real;
                T_cheg_armazenamento:real;
                T_entrada_fila:real;
                T_ordenacao:real;
                prioridade:byte;
                T_entrega_final:real;
                N_op_completar:byte;
```

```
                ordenador:real;
                T_set_up_lote:real;
                Tam_lote:word;
                saida:boolean;
                pos_atual:byte;
                maq:byte;
                ptr_proximo:ponteiro;
                end;
  Machine=record
                 T_set_up:real;
                 name:palavra;
                 Ident_maquina:byte;
                 situacao:status_maq;
                 cond_operacao:cond_operacao_maq;
                 cond_saturacao:byte;
                 distr_quebra:byte;
                 par_quebra1:real;
                 par_quebra2:real;
                 par_quebra3:real;
                 dist_reparo:byte;
                 par_rep1:real;
                 par_rep2:real;
                 par_rep3:real;
                 end;
  Manipulator=record
                nome:palavra;
                situacao:status_manip;
                Veloc:real;
                cond_operacao:cond_operacao_manip;
                posicao:byte;
                distr_quebra:byte;
                par_quebra1:real;
                par_quebra2:real;
                par_quebra3:real;
                dist_reparo:byte;
                par_rep1:real;
                par_rep2:real;
                par_rep3:real;
                end;
  ptr_celula=^cell;
  Cell=record
                Ident_celula:byte;
                N_maquinas:byte;
                maquina:array [1..5]of machine;
                manip:manipulator;
                ptr_proximo:ptr_celula;
                end;
  Sist_Transf=record
              tipo:palavra;
              velocidade:real;
              estado:status_manip;
              cond_operacao:cond_operacao_manip;
              posicao:byte;
              end;
operacao= (armazena,manipulador,maquina,SisTrans,quebra_maquina,reparo_maquina);
direcao=(para_maquina,para_ST,para_buffer);
direcao_ST=(para_armazenamento,para_celula);
ptr_evento= registro_evento;
registro_evento=record
                     T_ocorrencia:real;
                     ptr_proximo:ptr_evento;
                     case operador:operacao of
                     armazena:();
                     SisTrans:(Ident_ST:byte;direction:direcao_ST);
                     maquina:(Ident_maquina,Ident_cell,Indice_maq:byte);
                     manipulador:(Ident_atividade:direcao;Ident_celula,Indice_machine:byte);
                     quebra_maquina:(Maquina,Cell,Indice:byte);
                     reparo_maquina:(Identifica_maquina,Ident_Num_Cel,Indice_Mac:byte);
                     end;
Manipula=array [1..20] of ponteiro;
Working=array [1..50,1..5] of ponteiro;

var
   Pont_celula:array [1..20] of ponteiro;
   Pont_ST:ponteiro;
   Ocupa_ST:array [1..5] of ponteiro;
   raiz_celula:ptr_celula;
   Manip: Manipula;
   Work: Working;
   celula:ptr_celula;
```

```
        armazenamento:ponteiro;
        raiz:ptr_evento;
        evento:ptr_evento;
        ST:array [1..5] of Sist_Transf;
        posicao:byte;
        N_fila_ST:word;
        N_fila_Manipulador:array [1..20] of word;
        N_fila_Maq:array [1..50] of word;
        Maq:byte;
        Cel:byte;
        Grupos:byte;
        Num_grupo:array [1..50] of byte;
        Final:boolean;
        Ind_Maq:byte;
        T_simulacao:real;
        relogio:real;
        N_pecas_terminadas:integer;
        Num_celulas:byte;
        Num_maquinas:byte;
        Saida_Sist:ponteiro;
        Ident_Job:word;
        Num_tarefas:word;
        Num_componentes:byte;
        Raiz_job:ptr_job;
        Dist_saida:array [0..20] of real;
        Dist_Celulas:array [0..20,0..20] of real;
        Dist_InCelula:array [1..20,0..5,0..5] of real;
        Sist_prioridade:byte;
        N_equip_ST:byte;
        Saida_fila:boolean;
        Condicao_Saturacao:array [1..50] of byte;
        Estado_Saturacao:array [1..50] of byte;
        Deseja_quebra:byte;
        Output_Cel:Manipula;
        Output_Maq:Working;
        Existe:boolean;
        ArqEntrada:text;
        (Variaveis Estatisticas)

        T_util_manip:array [1..20] of real;
        T_buffer_manip:array [1..20] of real;
        T_ocupado_manip:array [1..20] of real;
        T_reparo_maq:array [1..20,1..5] of real;
        Perc_ocupado_manip:real;
        Perc_util_manip:real;
        Perc_buffer_manip:real;
        Perc_ocupacao_maq:real;
        Perc_reparo_maq:real;
        T_ocup_maq:array [1..20,1..5] of real;
        T_ult_alt_fila_ST:real;
        T_ult_alt_fila_maq:array [1..50] of real;
        T_ult_alt_fila_manip:array [1..50] of real;
        Area_sob_curva_ST:real;
        Area_sob_curva_manip:array [1..20] of real;
        Area_sob_curva_maq:array [1..50] of real;
        T_demora_fila:array [1..100] of real;
        N_vezes_fila:array [1..100] of longint;
        T_medio_espera:real;
        N_jobs_completos:array [1..100] of word;
        Num_medio_fila_manip:real;
        Num_medio_fila_maq:real;
        Num_medio_fila_ST:real;
        Numero:word;
        T_ocupado_ST:real;
        T_util_ST:real;
        T_armaz_ST:real;
        Perc_ocupado_ST:real;
        Perc_util_ST:real;
        Perc_armaz_ST:real;
Implementation
end.

Unit Varler;
Interface
Uses Dos,Crt,defvar;
     type
        sentenca=string[43];
     VAR
        Var2deArquivo:text;
        Nome_arquivo_coord:string[43];
```

```
        dist_saida_x,dist_saida_y:real;
        dist_celula:array [0..20,1..2] of real;
        coord_maq_incelula:array [1..20,0..5,1..2] of real;
        arq_rel:string[43];
        VardeArquivo:text;
        Nome_da_simulacao, Nome_maquina, Nome_manipulador:string[30];
        Nome_componente, Nome_sistema_transferencia:string[30];
        Nome_arquivo:string[43];
        drive_arquivo:string[30];
        contador,i,i1,i2,i3:byte;
        Cond_saturacao:byte;
        Numero_maquinas:array [1..20] of byte;
        N_maquina:byte;
        N_celulas:byte;
        N_componentes:word;
        Ident_componente,Ident_celula, Ident_maquina:byte;
        coluna,linha:byte;
        Tempo_setup, velocidade_manipulador, par_lote, par_opcao, velocidade_sistema_transferencia:real;
        Parametro1,Parametro2,Parametro3,var1,var2,var3:real;
        prioridade_componente:byte;
        resposta,resposta2,resp:string[3];
        F_prob_quebra,F_prob_reparo:byte;
        F_prob_tempo_chegada, N_operacoes, opcao_1_maquina, opcao_2_maquina, opcao_3_maquina:byte;
        sistema_prioridade,F_prob_tamanho_lote, Tamanho_lote, N_parametros:byte;
        F_prob_opcao_1,F_prob_opcao_2,F_prob_opcao_3:byte;
        Par11,Par12,Par13,Par21,Par22,Par23,Par31,Par32,Par33:real;
        N_equip_Sist_Transf,N_par1,N_par2,N_par3:byte;
        N_ppar2:byte;
        const1:string[16];
        const2:real;
        identificacao:byte;

Implementation

end.

Unit Varrer;
Interface
Uses dos,crt,defvar;
VAR
   Nome_arq_grav_rel:string[30];
   ArqRelat:TEXT;
   N_par_quebra,N_par_reparo:byte;
   rel_celula:ptr_celula;
   rel_job:ptr_job;
   rel_componente:ponteiro;
   N_par_cheg:byte;
   N_par_Tam_lote:byte;
   N_par_prob:byte;
Implementation

end.

Unit Exec;
Interface
Uses Overlay,Crt,Dos,Printer,learq,prior,tela,Defvar,prob,Lista_celula,
     Lista_evento,Lista_job,Lista_tarefa,Est,Util,Quebra,Saida,Comeca,
     relatres,relatori,varler;
{$O learq}
{$O prior}
{$O tela}
{$O coord}
{$O ler}
{$O relatori}
{$O relatres}

   procedure proximo_evento;
   procedure execucao;

Implementation

procedure proximo_evento;
begin
     del_prim_evento(raiz,evento);
     relogio:=evento^.T_ocorrencia;
     case evento^.operador of
     armazena:saida_armazenamento;
     SisTrans:saida_ST;
     manipulador:saida_manipulador;
     maquina:saida_maquina;
```

```
        quebra_maquina:quebra_de_maquina;
        reparo_maquina:reparo_de_maquina;
        end;
        FreeMem(evento,SizeOf(registro_evento));
    end;

    procedure execucao;
    begin
        clrscr;
        moldura(1,1,80,25);
        window(2,2,79,24);
        window(1,1,80,25);
        gotoxy(5,5);
        write('    TEMPO DE SIMULACAO =');
        readln(T_simulacao);
        entr_sistema_prioridade;
        leitura_arquivo;
        window(1,1,80,25);
        clrscr;
        moldura(1,1,80,25);
        entr_nome_arq_rel;
        inicio;
        inicializacao;
        clrscr;
        moldura(45,9,75,11);
        moldura(9,9,34,21);
        moldura(9,7,34,21);
        gotoxy(46,10);
        write('TEMPO DE SIMULACAO=',T_simulacao:7:2);
        gotoxy(15,8);
        write('*** RELOGIO ***');
        window(10,10,33,20);
        gotoxy(1,1);
        while relogio (= T_simulacao do
        begin
            writeln('       ',relogio:8:4);
            proximo_evento;
        end;
        relatorio(arq_rel);
        relatresult(arq_rel);
        window(1,1,80,25);
end;
end.

Unit Ocupa;
Interface
Uses Dos,Defvar,Lista_celula,Lista_tarefa,Lista_evento,Prob,Est,Util;

        procedure coloca_lista_celula (Maquina, N_Cel, Indice_Maq:byte; Cell:ptr_celula; Pont:ponteiro);
        procedure Ocupar_ST(Auxiliar:ponteiro;Cel_origem:byte);
        procedure ocupa_maquina(Cell:ptr_celula;Indice_maq,Machine:byte;Pontero:ponteiro);
        procedure ocupando_manipulador (N_Cel:byte; KK:ponteiro; Cell:ptr_celula;Ind_Maq_livre,Maq_livre:byte);
        procedure ocupa_manipulador (N_Cel:byte; KK:ponteiro; Cell:ptr_celula;Ind_Maq_livre,Maq_livre:byte);

Implementation

procedure coloca_lista_celula (Maquina, N_Cel, Indice_Maq:byte; Cell:ptr_celula; Pont:ponteiro);
var
   KK:ptr_celula;
   TT,LL,MM:byte;
   Nada:boolean;
   Ptr:ptr_evento;
   Subst:real;
begin
    Pont^.pos_atual:=N_Cel;
    if (Existe=true) and (Indice_Maq()0) and (Cell^.manip.situacao=nao_ocupado)
    then begin
        Cell^.manip.situacao:=ocup;
        Manip[N_Cel]:=Pont;
        GetMem(Ptr,SizeOf(registro_evento));
        if Cell^.manip.veloc () 0
        then Ptr^.T_ocorrencia:=relogio+(Dist_InCelula[N_Cel,0,Indice_Maq]/Cell^.manip.veloc)
        else Ptr^.T_ocorrencia:=relogio;
        T_ocupado_manip[N_Cel]:=T_ocupado_manip[N_Cel]+Ptr^.T_ocorrencia-relogio;
        T_util_manip[N_Cel]:=T_util_manip[N_cel]+Ptr^.T_ocorrencia-relogio;
        Ptr^.operador:=manipulador;
        Ptr^.Ident_celula:= N_Cel;
        Ptr^.Ident_atividade:= para_maquina;
        Ptr^.Indice_Machine:=Indice_maq;
        insere_evento (Raiz, Ptr);
```

```
          end
     else begin
          Pont^.maq:=0;
          Pont^.T_entrada_fila:=relogio;
          N_vezes_fila[Pont^.identificacao]:=N_vezes_fila[Pont^.identificacao]+1;
          insere_prioridade(Pont_Celula[N_Cel], Pont);
          Incrementa_fila_manipulador(N_Cel);
          Incrementa_fila_maquina(Maquina);
          satura(Maquina,Cell)
          end;
end;


procedure Ocupar_ST(Auxiliar:ponteiro;Cel_origem:byte);
var
   Ocupa,Ocupei:boolean;
   Pi:ptr_evento;
   H,N_Cel,M,pos:byte;
   Cell:ptr_celula;
   N_ST:byte;
   Nao_Achei:boolean;
   Auxiliar_novo:ponteiro;
begin
     Ocupa:=false;
     Ocupei:=true;
     N_ST:=1;
     Nao_Achei:=true;
     if Auxiliar^.saida=true
     then begin
          while (N_ST (= N_equip_ST) and (Nao_Achei) do
          begin
               if ST[N_ST].estado=nao_ocupado
               then begin
                    Ocupa_ST[N_ST]:=Auxiliar;
                    ST[N_ST].estado:=ocup;
                    Nao_Achei:=false;
                    Ocupei:=false;
                    GetMem(Pi,SizeOf(registro_evento));
                    if ST[N_ST].velocidade (> 0
                    then Pi^.T_ocorrencia:=relogio+(Dist_saida[Auxiliar^.pos_atual]/ST[N_ST].velocidade)
                    else Pi^.T_ocorrencia:=relogio;
                    T_ocupado_ST:=T_ocupado_ST+Pi^.T_ocorrencia-relogio;
                    T_util_ST:=T_util_ST+Pi^.T_ocorrencia-relogio;
                    Pi^.operador:=SisTrans;
                    Pi^.Ident_ST:=N_ST;
                    Pi^.direction:=para_celula;
                    insere_evento(raiz,Pi);
                    end;
               N_ST:=N_ST+1;
          end;
          end
     else begin
          while (N_ST (= N_equip_ST) and (Nao_Achei) do
          begin
               if ST[N_ST].estado=nao_ocupado
               then begin
                    Ocupa_ST[N_ST]:=Auxiliar;
                    ST[N_ST].estado:=ocup;
                    Nao_Achei:=false;
                    maquina_a_ser_ocupada(Auxiliar,H,N_Cel,M,Cell,Ocupa);
                    if Ocupa=true
                    then begin
                         Ocupei:=false;
                         GetMem(Pi,SizeOf(registro_evento));
                         if ST[N_ST].velocidade (> 0
                         then Pi^.T_ocorrencia:=relogio+(Dist_celulas[Auxiliar^.pos_atual,N_Cel]/ST[N_ST].velocidade)
                         else Pi^.T_ocorrencia:=relogio;
                         T_ocupado_ST:=T_ocupado_ST+Pi^.T_ocorrencia-relogio;
                         T_util_ST:=T_util_ST+Pi^.T_ocorrencia-relogio;
                         Pi^.operador:=SisTrans;
                         Pi^.Ident_ST:=N_ST;
                         Pi^.direction:=para_celula;
                         insere_evento(raiz,Pi);
                         end
                    else begin
                         Ocupa_ST[N_ST]^.operador:=0;
                         GetMem(Pi,SizeOf(registro_evento));
                         if ST[N_ST].velocidade (> 0
                         then Pi^.T_ocorrencia:=relogio+(Dist_celulas[0,Auxiliar^.pos_atual]/ST[N_ST].velocidade)
                         else Pi^.T_ocorrencia:=relogio;
```

```
                    T_ocupado_ST:=T_ocupado_ST+P1^.T_ocorrencia-relogio;
                    T_armaz_ST:=T_armaz_ST+P1^.T_ocorrencia-relogio;
                    P1^.operador:=SisTrans;
                    P1^.Ident_ST:=N_ST;
                    P1^.direction:=para_armazenamento;
                    Ocupei:=false;
                    insere_evento(raiz,P1);
                    end;
                end;
                N_ST:=N_ST+1;
        end;
        end;
    if Ocupei=true
    then begin
        if (Cel_origem <> 0) and (Output_Cel[Cel_Origem]=nil)
        then begin
            Output_Cel[Cel_origem]:=Auxiliar;
            end;
        Auxiliar^.T_entrada_fila:=relogio;
        N_vezes_fila[Auxiliar^.identificacao]:=N_vezes_fila[Auxiliar^.identificacao]+1;
        insere_prioridade(Pont_ST,Auxiliar);
        Incrementa_fila_ST;
        end;
end;

procedure ocupa_maquina(Cell:ptr_celula; Indice_Maq,Machine:byte;Pontero:ponteiro);
var
   Header:ptr_evento;
   Subst:real;
   N_cel:byte;
begin
    Cell^.maquina[Indice_Maq].situacao:=ocupado;
    Pontero^.maq:=Indice_Maq;
    N_cel:=Cell^.Ident_celula;
    T_ocup_maq[N_cel,Indice_maq]:= T_ocup_maq[N_cel,Indice_Maq]+Pontero^.T_ocupacao;
    GetMem(Header,SizeOf(registro_evento));
    Header^.T_ocorrencia:=relogio+Pontero^.T_ocupacao;
    Header^.operador:=maquina;
    Header^.Indice_maq:=Indice_Maq;
    Header^.Ident_maquina:=Machine;
    Header^.Ident_cell:=Cell^.Ident_celula;
    insere_evento(Raiz,Header);
end;

procedure ocupando_manipulador (N_Cel:byte; KN:ponteiro; Cell:ptr_celula;Ind_Maq_livre,Maq_livre:byte);
var
   P5,Ptr:ptr_evento;
   Maquina, Indice_Maq, Machine:byte;
   Subst:real;
   K:byte;
   T2:ponteiro;
   Ali:boolean;
begin
        if KN^.saida=true
        then begin
            Cell^.manip.situacao:=ocup;
            Incrementa_espera(KN);
            Manip[N_Cel]:=KN;
            GetMem (P5,SizeOf(registro_evento));
            if Cell^.manip.veloc <> 0
            then P5^.T_ocorrencia:=relogio+(Dist_InCelula[N_Cel,0,KN^.maq]/Cell^.manip.veloc)
            else P5^.T_ocorrencia:=relogio;
            T_ocupado_manip[N_Cel]:=T_ocupado_manip[N_cel]+P5^.T_ocorrencia-relogio;
            T_util_manip[N_Cel]:=T_util_manip[N_cel]+P5^.T_ocorrencia-relogio;
            P5^.operador:=manipulador;
            P5^.Ident_celula:=Cell^.Ident_celula;
            P5^.Ident_atividade:=para_ST;
            insere_evento(Raiz,P5);
            end
        else begin
            Maquina:=KN^.operador;
            encontra_maquina(Maquina,Cell,Indice_Maq);
            if (Indice_Maq=0) and (Existe=false)
            then begin
                Cell^.manip.situacao:=ocup;
                Incrementa_espera(KN);
                Manip[N_Cel]:=KN;
                GetMem(P5,SizeOf(registro_evento));
                if Cell^.manip.veloc <> 0
                then P5^.T_ocorrencia:=relogio+(Dist_InCelula[N_Cel,0,KN^.maq]/Cell^.manip.veloc)
```

```
                    else P5^.T_ocorrencia:=relogio;
                    T_ocupado_manip[N_Cel]:=T_ocupado_manip[N_cel]+P5^.T_ocorrencia-relogio;
                    T_util_manip[N_Cel]:=T_util_manip[N_cel]+P5^.T_ocorrencia-relogio;
                    P5^.operador:=manipulador;
                    P5^.Ident_celula:=Cell^.Ident_celula;
                    P5^.Ident_atividade:=para_ST;
                    insere_evento(Raiz,P5);
                    end
               else begin
                    if (Indice_Maq()0) and (Existe=true)
                    then begin
                         Cell^.manip.situacao:=ocup;
                         Incrementa_espera(KN);
                         Manip[N_Cel]:=KN;
                         GetMem(P5,SizeOf(registro_evento));
                         if Cell^.manip.veloc () 0
                         then P5^.T_ocorrencia:=relogio+(Dist_InCelula[N_Cel,KN^.maq,Indice_Maq]/Cell^.manip.veloc)
                         else P5^.T_ocorrencia:=relogio;
                         T_ocupado_manip[N_Cel]:=T_ocupado_manip[N_cel]+P5^.T_ocorrencia-relogio;
                         T_util_manip[N_Cel]:=T_util_manip[N_cel]+P5^.T_ocorrencia-relogio;
                         P5^.operador:=manipulador;
                         P5^.Ident_celula:=Cell^.Ident_celula;
                         P5^.Ident_atividade:=para_maquina;
                         P5^.Indice_Machine:=Indice_Maq;
                         insere_evento(Raiz,P5);
                         Decrementa_fila_maquina(maquina);
                         desatura(Maquina,Cell);
                         end
                    else begin
                         Cell^.manip.situacao:=ocup;
                         Manip[N_Cel]:=KN;
                         GetMem(P5,SizeOf(registro_evento));
                         P5^.operador:=manipulador;
                         if Cell^.manip.veloc () 0
                         then P5^.T_ocorrencia:=relogio+(Dist_InCelula[N_Cel,0,Ind_Maq_livre]/Cell^.manip.veloc)
                         else P5^.T_ocorrencia:=relogio;
                         T_ocupado_manip[N_Cel]:=T_ocupado_manip[N_cel]+P5^.T_ocorrencia-relogio;
                         T_buffer_manip[N_Cel]:=T_buffer_manip[N_cel]+P5^.T_ocorrencia-relogio;
                         P5^.Ident_atividade:=para_buffer;
                         P5^.Ident_celula:=Cell^.Ident_celula;
                         insere_evento(raiz,P5);
                         end;
               end;
          end;
end;

procedure ocupa_manipulador (N_Cel:byte; KN:ponteiro; Cell:ptr_celula;Ind_Maq_livre,Maq_livre:byte);
var
   P5,Ptr:ptr_evento;
   Maquina, Indice_Maq, Machine:byte;
   Subst:real;
   K:byte;
   T2:ponteiro;
   Ali:boolean;
begin
     if Cell^.manip.situacao=ocup
     then begin
          KN^.T_entrada_fila:=relogio;
          N_vezes_fila[KN^.identificacao]:=N_vezes_fila[KN^.identificacao]+1;
          if (Ind_maq_livre () 0) and (Output_Maq[N_Cel,Ind_Maq_livre]=nil)
              then Output_Maq[N_Cel,Ind_Maq_livre]:=KN;
          insere_prioridade(Pont_celula[N_Cel],KN);
          Incrementa_fila_manipulador(N_Cel);
          if (KN^.saida = false)
          then begin
               K:=KN^.operador;
               Maq_na_Celula(K,Cell,Ali);
               if Ali=true
               then begin
                    Incrementa_fila_maquina(K);
                    satura(K,Cell);
                    end;
               end;
          end
     else begin
          if KN^.saida=true
          then begin
               Cell^.manip.situacao:=ocup;
               Manip[N_Cel]:=KN;
               GetMem (P5,SizeOf(registro_evento));
```

```
                if Cell^.manip.veloc () 0
                then P5^.T_ocorrencia:=relogio+(Dist_InCelula[N_Cel,0,KN^.maq]/Cell^.manip.veloc)
                else P5^.T_ocorrencia:=relogio;
                T_ocupado_manip[N_Cel]:=T_ocupado_manip[N_cel]+P5^.T_ocorrencia-relogio;
                T_util_manip[N_Cel]:=T_util_manip[N_cel]+P5^.T_ocorrencia-relogio;
                P5^.operador:=manipulador;
                P5^.Ident_celula:=Cell^.Ident_celula;
                P5^.Ident_atividade:=para_ST;
                insere_evento(Raiz,P5);
                end
            else begin
                Maquina:=KN^.operador;
                encontra_maquina(Maquina,Cell,Indice_Maq);
                if (Indice_Maq=0) and (Existe=false)
                then begin
                    Cell^.manip.situacao:=ocup;
                    ManipN_Cel]:=KN;
                    GetMem(P5,SizeOf(registro_evento));
                    if Cell^.manip.veloc () 0
                    then P5^.T_ocorrencia:=relogio+(Dist_InCelula[N_Cel,0,KN^.maq]/Cell^.manip.veloc)
                    else P5^.T_ocorrencia:=relogio;
                    T_ocupado_manip[N_Cel]:=T_ocupado_manip[N_cel]+P5^.T_ocorrencia-relogio;
                    T_util_manip[N_Cel]:=T_util_manip[N_cel]+P5^.T_ocorrencia-relogio;
                    P5^.operador:=manipulador;
                    P5^.Ident_celula:=Cell^.Ident_celula;
                    P5^.Ident_atividade:=para_ST;
                    insere_evento(Raiz,P5);
                    end
                else begin
                    if (Indice_Maq()0) and (Existe=true)
                    then begin
                        Cell^.manip.situacao:=ocup;
                        ManipN_Cel]:=KN;
                        GetMem(P5,SizeOf(registro_evento));
                        if Cell^.manip.veloc () 0
                        then P5^.T_ocorrencia:=relogio+(Dist_InCelula[N_Cel,KN^.maq,Indice_Maq]/Cell^.manip.veloc)
                        else P5^.T_ocorrencia:=relogio;
                        T_ocupado_manip[N_Cel]:=T_ocupado_manip[N_cel]+P5^.T_ocorrencia-relogio;
                        T_util_manip[N_Cel]:=T_util_manip[N_cel]+P5^.T_ocorrencia-relogio;
                        P5^.operador:=manipulador;
                        P5^.Ident_celula:=Cell^.Ident_celula;
                        P5^.Ident_atividade:=para_maquina;
                        P5^.Indice_Machine:=Indice_Maq;
                        insere_evento(Raiz,P5);
                        end
                    else begin
                        Cell^.manip.situacao:=ocup;
                        ManipN_Cel]:=KN;
                        GetMem(P5,SizeOf(registro_evento));
                        P5^.operador:=manipulador;
                        if Cell^.manip.veloc () 0
                        then P5^.T_ocorrencia:=relogio+(Dist_InCelula[N_Cel,0,Ind_Maq_livre]/Cell^.manip.veloc)
                        else P5^.T_ocorrencia:=relogio;
                        T_ocupado_manip[N_Cel]:=T_ocupado_manip[N_cel]+P5^.T_ocorrencia-relogio;
                        T_buffer_manip[N_Cel]:=T_buffer_manip[N_cel]+P5^.T_ocorrencia-relogio;
                        P5^.Ident_atividade:=para_buffer;
                        P5^.Ident_celula:=Cell^.Ident_celula;
                        insere_evento(raiz,P5);
                        Incrementa_fila_maquina(Maquina);
                        satura(Maquina,Cell);
                        end;
                    end;
                end;
            end;
        end;
end;

end. (Fim da Unit)

Unit Util;
Interface
Uses Dos,Defvar,Lista_celula,Lista_tarefa,Lista_evento,Lista_job,Prob,Est;

    procedure troca_ponteiros(Velho:ponteiro; var Novo:ponteiro);
    procedure gera_novo_el_armazenamento(KK:ponteiro; Pont:ponteiro);
    procedure maquina_a_ser_ocupada(ponte:ponteiro;var Maquina,N_Cel,Indice_Maq:byte; var Cell:ptr_celula; var fila:boolean);
    procedure proxima_operacao(Temp:ponteiro; var Xend:boolean);
    procedure encontra_maq_lista_celula(chave:byte;prim:ponteiro;var Pont:ponteiro);
    procedure encontra_buffer_cel(chave:byte;prim:ponteiro;var Pont:ponteiro);
    procedure percorre_lista_ST(Pont:ponteiro;var Retorno:ponteiro;var N_Cel:byte;var Codigo:boolean);
    procedure percorre_lista_celula(Pont:Ponteiro; Cell: ptr_celula; var KK:ponteiro; var Indice:byte);
```

```
Implementation

procedure  troca_ponteiros(Velho:ponteiro; var Novo:ponteiro);
var
   X:real;
   I:byte;
begin
     GetMem(Novo,SizeOf(tarefa));
     Novo^.identificacao:=Velho^.identificacao;
     Novo^.rota:=1;
     Novo^.Tam_lote:=Velho^.Tam_lote;
     Novo^.prioridade:=Velho^.prioridade;
     Novo^.N_op_completar:=Velho^.N_op_completar;
     Novo^.T_entrega_final:=Velho^.T_entrega_final;
     Novo^.T_set_up_lote:=Velho^.T_set_up_lote;
     Novo^.ordenador:=Velho^.ordenador;
     Novo^.saida:=false;
     Novo^.pos_atual:=Velho^.pos_atual;
     Novo^.maq:=Velho^.maq;
     Novo^.T_ocupacao:=0.0;
     Novo^.operador:=0;
     Novo^.maq:=0;
     Novo^.pos_atual:=0;
end;

procedure gera_novo_el_armazenamento(KK:ponteiro; Pont:ponteiro);
var
   Procura:ptr_job;
   Comp:byte;
   Tempo:real;
   Tamanho:word;
begin
     Comp:=Pont^.identificacao;
     encontra_job(Comp,raiz_job,Procura);
     tempo_de_chegada(Procura,Tempo);
     Pont^.T_cheg_armazenamento:=relogio+Tempo;
     Pont^.T_ordenacao:=Pont^.T_cheg_armazenamento;
     tamanho_de_lote(Procura,Tamanho);
     Pont^.Tam_lote:=Tamanho;
     cria_lista_ordenada(KK,pont);
end;

procedure maquina_a_ser_ocupada(ponte:ponteiro;var Maquina,N_Cel,Indice_Maq:byte; var Cell:ptr_celula; var fila:boolean);
var
   Achei,Code,Vai:boolean;
   Cont,Aux,Comp,Ind,IndiceMaq:byte;
   Procura:ptr_job;
   KCel:ptr_celula;
begin
     Achei:=true;
     Code:=true;
     fila:=false;
     Cont:=1;
     Comp:=ponte^.identificacao;
     Ind:=ponte^.rota;
     encontra_job(Comp,Raiz_job,Procura);
     while (Achei) and (Cont(=3) do
     begin
          Aux:=Procura^.elemento[Ind].maquina[Cont];
          if Aux=0
          then begin
               Achei:=false;
               fila:=false;
               end
          else begin
               encontra_celula(Aux,Raiz_celula,KCel,Indice_Maq);
               if (Indice_Maq () 0) and Existe=true
               then begin
                    if (KCel^.manip.cond_operacao=rotina)
                    then begin
                         ponte^.operador:=Aux;
                         Cell:=KCel;
                         N_Cel:=Cell^.Ident_celula;
                         Maquina:=Aux;
                         escolhe_funcao_probabilidade(Procura,ponte,Ind,Cont);
                         Achei:=false;
                         Code:=false;
                         fila:=true;
                         end;
```

```
                    end;
                end;
            Cont:=Cont+1;
        end;
        if Code=true
        then begin
            Cont:=1;
            Achei:=true;
            fila:=false;
            while (Achei) and (Cont(=3) do
            begin
                Aux:=Procura^.elemento[Ind].maquina[Cont];
                if Aux=0
                then begin
                    Achei:=false;
                    fila:=false;
                    end
                else begin
                    encontra_maquina_celula(Aux,Raiz_celula,KCel,IndiceMaq,Vai);
                    if Vai=true
                    then begin
                        if (KCel^.manip.cond_operacao=rotina)
                        then begin
                            ponte^.operador:=Aux;
                            N_Cel:=KCel^.Ident_celula;
                            Maquina:=Aux;
                            escolhe_funcao_probabilidade(Procura,ponte,Ind,Cont);
                            Achei:=false;
                            fila:=true;
                            end;
                        end;
                    end;
                Cont:=Cont+1;
            end;
        end;
end;

procedure proxima_operacao(Temp:ponteiro; var Xend:boolean);
var
   Comp:byte;
   Procura:ptr_job;
   TT,LL,NN:byte;   (variaveis de compatibilizacao de rotinas)
   TN2:ptr_celula;  (idem)
   Deleta:boolean;  (idem)
begin
     Xend:=False;
     Comp:=Temp^.identificacao;
     encontra_job(Comp,Raiz_job,Procura);
     Temp^.rota:=Temp^.rota+1;
     if (Temp^.rota=Procura^.last)
     then begin
         Xend:=true;
         Temp^.operador:=0;
         end
     else begin
         Temp^.N_op_completar:= Temp^.N_op_completar-1;
         maquina_a_ser_ocupada(Temp,TT,LL,NN,TN2,deleta);
         if Deleta=false then Temp^.operador:=0;
         end;
end;

procedure encontra_maq_lista_celula(chave:byte;prim:ponteiro;var Pont:ponteiro);
var
   Tempor:ponteiro;
   Nao_Achei:boolean;
begin
     Pont:=nil;
     Tempor:=prim^.ptr_proximo;
     Nao_Achei:=true;
     while (Tempor()nil) and Nao_Achei do
     begin
         if (Tempor^.operador=chave)
         then begin
             Nao_Achei:=false;
             Pont:=Tempor;
             end;
         Tempor:=Tempor^.ptr_proximo;
     end;
end;
```

```
        procedure encontra_buffer_cel(chave:byte;prim:ponteiro;var Pont:ponteiro);
        var
           Tempor:ponteiro;
           Nao_Achei:boolean;
        begin
             Pont:=nil;
             Tempor:=prim^.ptr_proximo;
             Nao_Achei:=true;
             while (Tempor()nil) and Nao_Achei do
             begin
                  if (Tempor^.pos_atual=chave)
                  then begin
                       Nao_Achei:=false;
                       Pont:=Tempor;
                       end;
                  Tempor:=Tempor^.ptr_proximo;
           end;
      end;

      procedure percorre_lista_ST(Pont:ponteiro;var Retorno:ponteiro;var N_Cel:byte;var Codigo:boolean);
      var
         Nao_Achei:boolean;
         MM,TT:byte;
         Cell:ptr_celula;
         Deleta:boolean;
         Prov:ponteiro;
      begin
          Nao_Achei:=true;
          N_Cel:=0;
          Retorno:=nil;
          Prov:=Pont^.ptr_proximo;
          while (Nao_Achei) and (Prov () nil) do
          begin
              if (Prov^.saida=true)
              then begin
                   Nao_Achei:=false;
                   Retorno:=Prov;
                   end
              else begin
                   maquina_a_ser_ocupada(Prov,MM,N_Cel,TT,Cell,Deleta);
                   if Deleta=true
                   then begin
                        Nao_achei:=false;
                        Retorno:=Prov;
                        end
                   else begin
                        if Prov^.pos_atual ()0
                        then begin
                             Prov^.operador:=0;
                             Nao_Achei:=false;
                             Retorno:=Prov;
                             Codigo:=true;
                             end;
                        end;
                   end;
          Prov:=Prov^.ptr_proximo;
      end;
end;

procedure percorre_lista_celula(Pont:Ponteiro; Cell: ptr_celula; var KK:ponteiro; var Indice:byte);
var
  Achei, Nao_Achei: boolean;
  K:byte;
  Temp_1:ponteiro;
begin
    Indice:=0;
    Achei:=true;
    KK:=nil;
    Temp_1:=Pont^.ptr_proximo;
    while (Temp_1 () nil) and Achei do
    begin
        if (Temp_1^.saida = true)
        then begin
             Achei:=false;
             Indice:=0;
             KK:=Temp_1;
             Existe:=false;
             end
        else begin
             K:=Temp_1^.operador;
```

```
                encontra_maquina (K,Cell,Indice);
                if (Indice=0) and (Existe=false)
                then begin
                     Achei:=false;
                     KK:=Temp_1;
                     end;
                if (Indice=0) and (Existe=true)
                then begin
                     if (temp_1^.maq <> 0)
                     then begin
                          Achei:=false;
                          KK:=Temp_1;
                          end;
                     end;
                if (Indice <> 0) and (Existe=true)
                then begin
                     Achei:=false;
                     KK:=Temp_1;
                     end;
                end;
           Temp_1:=Temp_1^.ptr_proximo;
      end;
end;

end. (Fim da Unit)

Unit Saida;
Interface
Uses Dos,Defvar,Lista_celula,Lista_tarefa,Lista_evento,Lista_job,Comeca,
     Ocupa,Util,Prob,Est;

     procedure saida_sistema(Temp_2:ponteiro);
     procedure saida_armazenamento;
     procedure saida_ST;
     procedure saida_manipulador;
     procedure saida_maquina;

Implementation

procedure saida_sistema(Temp_2:ponteiro);
begin
    Ident_Job:=Temp_2^.identificacao;
    N_jobs_completos[Ident_Job]:=N_jobs_completos[Ident_Job]+1;
    N_pecas_terminadas:=N_pecas_terminadas+1;
end;

procedure saida_armazenamento;
var
   TX1:ponteiro;
   T1:ponteiro;
begin
     del_prim_el(armazenamento,TX1);
     troca_ponteiros(TX1,T1);
     gera_novo_el_armazenamento(armazenamento,T1);
     primeiro_elemento(armazenamento,raiz);
     TX1^.T_ordenacao:=relogio;
     ocupar_ST(TX1,0);
end;

procedure Saida_ST;
var
   Sair,Ocupei,Para_Ocupar:boolean;
   T2:ptr_evento;
   KST,Temp,Aux,Aux1,Oc_Manip:ponteiro;
   N_ST:byte;
   N_celula,N_Cel,HH,MM:byte;
   Encontrei:boolean;
   Cell:ptr_celula;
   R:real;
begin
     N_ST:=evento^.Ident_ST;
     Ocupei:=true;
     Encontrei:=false;
     if evento^.direction=para_celula
     then begin
          if Ocupa_ST[N_ST]^.saida=true
          then begin
               Saida_Sist:=Ocupa_ST[N_ST];
               Ocupa_ST[N_ST]:=nil;
               saida_sistema(Saida_sist);
```

```
          FreeMem(Saida_Sist,SizeOf(tarefa));
          STCN_ST].estado:=nao_ocupado;
          end
       else begin
          Maq:=Ocupa_STCN_ST]^.operador;
          encontra_celula(Maq,Raiz_celula,Celula,Ind_Maq);
          Cel:=Celula^.Ident_celula;
          Oc_Manip:=Ocupa_STCN_ST];
          Ocupa_ST[N_ST]:=nil;
          Oc_Manip^.T_ordenacao:=relogio;
          coloca_lista_celula(Maq,Cel,Ind_Maq,Celula,Oc_Manip);
          STCN_ST].estado:=nao_ocupado;
          if (Output_Cel[Cel] <> nil)
          then begin
              Ocupei:=false;
              delecao(Pont_ST,Output_Cel[Cel]);
              Incrementa_espera(Ocupa_STCN_ST]);
              Ocupa_STCN_ST]:=Output_Cel[Cel];
              Ocupa_STCN_ST]^.T_ordenacao:=relogio;
              Output_Cel[Cel]:=nil;
              Decrementa_fila_ST;
              STCN_ST].estado:=ocup;
              encontra_buffer_cel(Cel,Pont_ST,Aux1);
              if Aux1<>nil
                 then Output_Cel[Cel]:=Aux1
                 else Output_Cel[Cel]:=nil;
              if (Ocupa_STCN_ST]^.saida=true)
              then begin
                   GetMem(T2,SizeOf(registro_evento));
                   if STCN_ST].velocidade <> 0
                   then T2^.T_ocorrencia:=relogio+(Dist_Saida[Ocupa_STCN_ST]^.pos_atual]/STCN_ST].velocidade)
                   else T2^.T_ocorrencia:=relogio;
                   T_ocupado_ST:=T_Ocupado_ST+T2^.T_ocorrencia-relogio;
                   T_util_ST:=T_util_ST+T2^.T_ocorrencia-relogio;
                   T2^.direction:=para_celula;
                   T2^.Ident_ST:=N_ST;
                   T2^.operador:=SisTrans;
                   insere_evento(raiz,T2);
                   end
              else begin
                   maquina_a_ser_ocupada(Ocupa_STCN_ST],MM,N_Cel,MM,Cell,Para_Ocupar);
                   if Para_Ocupar=true
                   then begin
                        GetMem(T2,SizeOf(registro_evento));
                        R:=STCN_ST].velocidade;
                        if STCN_ST].velocidade <> 0
                        then T2^.T_ocorrencia:=relogio+(Dist_celulas[Ocupa_STCN_ST]^.pos_atual,N_Cel]/R)
                        else T2^.T_ocorrencia:=relogio;
                        T_ocupado_ST:=T_ocupado_ST+T2^.T_ocorrencia-relogio;
                        T_util_ST:=T_util_ST+T2^.T_ocorrencia-relogio;
                        T2^.operador:=SisTrans;
                        T2^.Ident_ST:=N_ST;
                        T2^.direction:=para_celula;
                        insere_evento(raiz,T2);
                        end
                   else begin
                        Ocupa_STCN_ST]^.operador:=0;
                        GetMem(T2,SizeOf(registro_evento));
                        if STCN_ST].velocidade <> 0
                        then T2^.T_ocorrencia:=relogio+(Dist_celulas[0,Ocupa_STCN_ST]^.pos_atual]/STCN_ST].velocidade)
                        else T2^.T_ocorrencia:=relogio;
                        T_ocupado_ST:=T_ocupado_ST+T2^.T_ocorrencia-relogio;
                        T_armaz_ST:=T_armaz_ST+T2^.T_ocorrencia-relogio;
                        T2^.operador:=SisTrans;
                        T2^.Ident_ST:=N_ST;
                        T2^.direction:=para_armazenamento;
                        insere_evento(raiz,T2);
                        end;
                   end;
              end;
          end;
    end;
if evento^.direction=para_armazenamento
then begin
    Cel:=evento^.Ident_Celula;
    Temp:=Ocupa_STCN_ST];
    Ocupa_STCN_ST]:=nil;
    Temp^.pos_atual:=0;
  Temp^.T_ordenacao:=relogio;
    Temp^.T_entrada_fila:=relogio;
```

```
          N_vezes_fila[Temp^.identificacao]:=N_vezes_fila[Temp^.identificacao]+1;
          insere_prioridade(Pont_ST,Temp);
          Incrementa_fila_ST;
          ST[N_ST].estado:=nao_ocupado;
          end;
     if (N_fila_ST () 0) and Ocupei
     then begin
          percorre_lista_ST(Pont_ST,KST,N_Celula,Encontrei);
          if KST () nil
          then begin
               delecao(Pont_ST,KST);
               if (KST^.pos_atual () 0)
               then begin
                    if (KST=Output_Cel[KST^.pos_atual])
                    then encontra_buffer_cel(KST^.pos_atual,Pont_ST,Aux);
                         if Aux()nil then Output_Cel[KST^.pos_atual]:=Aux
                                     else Output_Cel[KST^.pos_atual]:=nil;
                    end;
               KST^.T_ordenacao:=relogio;
               Decrementa_fila_ST;
               Incrementa_espera(KST);
               Ocupa_ST[N_ST]:=KST;
               ST[N_ST].estado:=ocup;
               GetMem(T2,SizeOf(registro_evento));
               if Encontrei=false
               then begin
                    if (KST^.saida=true)
                    then begin
                         if ST[N_ST].velocidade () 0
                         then T2^.T_ocorrencia:=relogio+(Dist_saida[KST^.pos_atual]/ST[N_ST].velocidade)
                         else T2^.T_ocorrencia:=relogio;
                         end
                    else begin
                         if ST[N_ST].velocidade () 0
                         then T2^.T_ocorrencia:=relogio+(Dist_Celulas[KST^.pos_atual,N_Celula]/ST[N_ST].velocidade)
                         else T2^.T_ocorrencia:=relogio;
                         end;
                    T_ocupado_ST:=T_ocupado_ST+T2^.T_ocorrencia-relogio;
                    T_util_ST:=T_util_ST+T2^.T_ocorrencia-relogio;
                    T2^.operador:=SisTrans;
                    T2^.Ident_ST:=N_ST;
                    T2^.direction:=para_celula;
                    insere_evento(raiz,T2);
                    end
               else begin
                    T2^.operador:=SisTrans;
                    T2^.Ident_ST:=N_ST;
                    T2^.direction:=para_armazenamento;
                    if ST[N_ST].velocidade () 0
                    then T2^.T_ocorrencia:=relogio+Dist_Celulas[0,KST^.pos_atual]/ST[N_ST].velocidade
                    else T2^.T_ocorrencia:=relogio;
                    T_ocupado_ST:=T_ocupado_ST+T2^.T_ocorrencia-relogio;
                    T_armaz_ST:=T_armaz_ST+T2^.T_ocorrencia-relogio;
                    insere_evento(raiz,T2);
                    end;
               end;
          end;
end;

procedure saida_manipulador;
var
   Teste:byte;
   TT,LL,NN:byte;
   PT2:ptr_celula;
   Sair:boolean;
   Oc_maq,Oc_st,Temp,T2:ponteiro;
   Ptr:ptr_evento;
   Subst:real;
   Machine:byte;
   Ocupei,Desocupa:boolean;
begin
     Ocupei:=true;
     if evento^.Ident_atividade=para_maquina
     then begin
          Ind_maq:=evento^.Indice_machine;
          Oc_Maq:=Manip[Cel];
          Manip[Cel]:=nil;
          Oc_maq^.T_ordenacao:=relogio;
          Maq:=Oc_maq^.operador;
          encontra_so_celula(Cel,Raiz_celula,Celula);
```

```
     Celula^.manip.situacao:= nao_ocupado;
     Work[Maq,Ind_maq]:=Oc_Maq;
     ocupa_maquina(Celula,Ind_Maq,Maq,Work[Maq,Ind_Maq]);
     if Output_Maq[Cel,Ind_Maq] <> nil
     then begin
          delecao(Pont_Celula[Cel],Output_Maq[Cel,Ind_Maq]);
          Temp:=Output_Maq[Cel,Ind_Maq];
          Temp^.T_ordenacao:=relogio;
          Output_Maq[Cel,Ind_maq]:=nil;
          Decrementa_fila_manipulador(Cel);
          {if (Temp^.saida=false)
          then begin
               maquina_a_ser_ocupada(Temp,TT,LL,NN,PT2,Sair);
               if Sair=false then Temp^.operador:=0;
               end;}
          ocupando_manipulador(Cel,Temp,Celula,Ind_Maq,Maq);
          Ocupei:=false;
          end;
     end;
if evento^.Ident_atividade = para_ST
then begin
     Cel:=evento^.Ident_celula;
     encontra_so_celula(Cel,Raiz_celula,Celula);
     Oc_ST:=Manip[Cel];
     Oc_ST^.T_ordenacao:=relogio;
     Oc_ST^.maq:=0;
     Manip[Cel]:=nil;
     ocupar_ST(Oc_ST,Cel);
     Celula^.manip.situacao:=nao_ocupado;
     end;
if evento^.Ident_atividade=para_buffer
then begin
     Cel:=evento^.Ident_celula;
     encontra_so_celula(Cel,raiz_celula,Celula);
     Temp:=Manip[Cel];
     Temp^.T_ordenacao:=relogio;
     Manip[Cel]:=nil;
     Temp^.maq:=0;
     Celula^.manip.situacao:=nao_ocupado;
     insere_prioridade(Pont_celula[Cel],Temp);
     Incrementa_fila_manipulador(Cel);
     end;
 if (N_fila_manipulador[Cel] <> 0) and Ocupei
 then begin
      percorre_lista_celula (Pont_Celula[Cel],Celula,T2,Ind_Maq);
      if (T2 <> nil) and (Ind_Maq <> 0)
      then begin
           delecao(Pont_Celula[Cel],T2);
           T2^.T_ordenacao:=relogio;
           if T2^.maq <> 0
           then begin
                if (T2=Output_Maq[Cel,T2^.maq]) then Output_Maq[Cel,T2^.maq]:=nil;
                end;
           Manip[Cel]:=T2;
           Incrementa_espera(T2);
           Celula^.manip.situacao:=ocup;
           GetMem(Ptr,SizeOf(registro_evento));
           if Celula^.manip.veloc <> 0
           then Ptr^.T_ocorrencia:=relogio+(Dist_InCelula[Cel,T2^.maq,Ind_maq]/Celula^.manip.veloc)
           else Ptr^.T_ocorrencia:=relogio;
           T_ocupado_manip[Cel]:=T_ocupado_manip[Cel]+Ptr^.T_ocorrencia-relogio;
           T_util_manip[Cel]:=T_util_manip[Cel]+Ptr^.T_ocorrencia-relogio;
           Ptr^.operador:=manipulador;
           Ptr^.Ident_celula:=Celula^.Ident_celula;
           Ptr^.Ident_atividade:=para_maquina;
           Ptr^.Indice_Machine:=Ind_Maq;
           insere_evento(Raiz,Ptr);
           Decrementa_fila_manipulador(Cel);
           Machine:=Manip[Cel]^.operador;
           if N_fila_maq[Machine] > 0
           then begin
                Decrementa_fila_maquina(Machine);
                desatura(Machine,Celula);
                end;
           end;
          if (Ind_Maq=0) and (T2 <> nil) and (Existe=false)
          then begin
               delecao(Pont_Celula[Cel],T2);
               T2^.T_ordenacao:=relogio;
               Manip[Cel]:=T2;
```

```
                    if T2^.maq () 0
                    then begin
                         if (T2=Output_Maq[Cel,T2^.maq]) then Output_Maq[Cel,T2^.maq]:=nil;
                         end;
                    Celula^.manip.situacao:=ocup;
                    Incrementa_espera(T2);
                    GetMem (Ptr,SizeOf(registro_evento));
                    if Celula^.manip.veloc () 0
                    then Ptr^.T_ocorrencia:=relogio+(Dist_InCelula[Cel,0,T2^.maq]/Celula^.manip.veloc)
                    else Ptr^.T_ocorrencia:=relogio;
                    T_ocupado_manip[Cel]:=T_ocupado_manip[Cel]+Ptr^.T_ocorrencia-relogio;
                    T_util_manip[Cel]:=T_util_manip[Cel]+Ptr^.T_ocorrencia-relogio;
                    Ptr^.operador:=manipulador;
                    Ptr^.Ident_celula:=Celula^.Ident_celula;
                    Ptr^.Ident_atividade:=para_ST;
                    insere_evento(Raiz,Ptr);
                    Decrementa_fila_manipulador(Cel);
                    end;
               if (Ind_Maq=0) and (T2 () nil) and (Existe=true)
               then begin
                    delecao(Pont_Celula[Cel],T2);
                    T2^.T_ordenacao:=relogio;
                    Manip[Cel]:=T2;
                    if T2^.maq () 0
                    then begin
                         if (T2=Output_Maq[Cel,T2^.maq]) then Output_Maq[Cel,T2^.maq]:=nil;
                         end;
                    Celula^.manip.situacao:=ocup;
                    GetMem (Ptr,SizeOf(registro_evento));
                    if Celula^.manip.veloc () 0
                    then Ptr^.T_ocorrencia:=relogio+(Dist_InCelula[Cel,0,T2^.maq]/Celula^.manip.veloc)
                    else Ptr^.T_ocorrencia:=relogio;
                    T_ocupado_manip[Cel]:=T_ocupado_manip[Cel]+Ptr^.T_ocorrencia-relogio;
                    T_buffer_manip[Cel]:=T_buffer_manip[Cel]+Ptr^.T_ocorrencia-relogio;
                    Ptr^.operador:=manipulador;
                    Ptr^.Ident_celula:=Celula^.Ident_celula;
                    Ptr^.Ident_atividade:=para_buffer;
                    insere_evento(Raiz,Ptr);
                    Decrementa_fila_manipulador(Cel);
                    end;
               if T2 = nil
               then begin
                    Celula^.manip.situacao:=nao_ocupado;
                    end;
               end;
end;

procedure saida_maquina;
var
   Temporario:ponteiro;
begin
     Maq:=evento^.Ident_maquina;
     Cel:=evento^.Ident_cell;
     Ind_maq:=evento^.Indice_maq;
     Temporario:=Work[Maq,Ind_Maq];
     Work[Maq,Ind_Maq]:=nil;
     Temporario^.T_ordenacao:=relogio;
     encontra_so_celula(Cel,Raiz_celula,Celula);
     Celula^.Maquina[Ind_Maq].situacao:=desocupado;
     proxima_operacao(Temporario,Final);
     if (Final=true) then Temporario^.saida:=true;
     ocupa_manipulador(Cel,Temporario,Celula,Ind_Maq,Maq);
end;

end. (Fim da Unit)
```

```
Unit Quebra;
Interface
Uses Dos,Defvar,Prob,Est,Ocupa,Lista_evento,Lista_celula,Lista_tarefa,Lista_job,Util;

     procedure Descarrega_ST(Maquina:byte);
     procedure gera_quebra_maquina(Machine,Indice_Maq:byte;TN:ptr_celula);
     procedure satura_outras_maquinas (Cel:ptr_celula;Indice, Ident:byte);
     procedure desatura_outras_maquinas (Cel:ptr_celula;Indice, Ident:byte);
     procedure gera_reparo_maq(Maquina:byte;Num_Cel,Indice_Maq:byte; TN:ptr_celula);
     procedure quebra_de_maquina;
     procedure reparo_de_maquina;
Implementation

procedure Descarrega_ST(Maquina:byte);
var
   N_ST,N_celula,TT,XX:byte;
   Ptr,Pont:ptr_evento;
   Cel_2:ptr_celula;
   Deleta:boolean;
begin
     N_ST:=1;
     while (N_ST (= N_equip_ST) do
     begin
          if Ocupa_ST[N_ST] () nil
          then begin
               if Ocupa_ST[N_ST]^.operador=Maquina
               then begin
                    encontra_evento_ST(N_ST,raiz,Pont);
                    if Pont () nil
                    then begin
                         deleta_evento(raiz,Pont);
                         FreeMem(Pont,SizeOf(registro_evento));
                         end;
                    maquina_a_ser_ocupada(Ocupa_ST[N_ST],TT,N_celula,XX,Cel_2,Deleta);
                    if Deleta=true
                    then begin
                         GetMem(Ptr,sizeOf(registro_evento));
                         Ptr^.operador:=SisTrans;
                         Ptr^.Ident_ST:=N_ST;
                         Ptr^.direction:=para_celula;
                         if ST[N_ST].velocidade () 0
                         then Ptr^.T_ocorrencia:=relogio+Dist_Celulas[Ocupa_ST[N_ST]^.pos_atual,N_celula]/ST[N_ST].velocidade
                         else Ptr^.T_ocorrencia:=relogio;
                         T_ocupado_ST:=T_ocupado_ST+Ptr^.T_ocorrencia-relogio;
                         T_util_ST:=T_util_ST+Ptr^.T_ocorrencia-relogio;
                         insere_evento(raiz,Ptr);
                         end
                    else begin
                         Ocupa_ST[N_ST]^.operador:=0;
                         GetMem(Ptr,SizeOf(registro_evento));
                         Ptr^.operador:=SisTrans;
                         Ptr^.direction:=para_armazenamento;
                         Ptr^.Ident_ST:=N_ST;
                         if ST[N_ST].velocidade () 0
                         then Ptr^.T_ocorrencia:=relogio+Dist_Celulas[Ocupa_ST[N_ST]^.pos_atual,0]/ST[N_ST].velocidade
                         else Ptr^.T_ocorrencia:=relogio;
                         T_ocupado_ST:=T_ocupado_ST+Ptr^.T_ocorrencia-relogio;
                         T_armaz_ST:=T_armaz_ST+Ptr^.T_ocorrencia-relogio;
                         insere_evento(raiz,Ptr);
                         end;
                    end;
               end;
          N_ST:=N_ST+1;
     end;
end;

procedure gera_quebra_maquina(Machine,Indice_Maq:byte;TN:ptr_celula);
var
   T_quebra:real;
   Quebra:ptr_evento;
begin
     GetMem(Quebra,SizeOf(registro_evento));
     Quebra^.operador:=quebra_maquina;
     Quebra^.Cell:=TN^.Ident_Celula;
     Quebra^.Indice:=Indice_Maq;
     Quebra^.Maquina:=Machine;
     tempo_quebra(TN,Indice_maq,T_quebra);
     Quebra^.T_ocorrencia:=relogio+T_quebra;
     insere_evento(raiz,Quebra);
end;
```

```
procedure satura_outras_maquinas (Cel:ptr_celula;Indice, Ident:byte);
var
   I:byte;
begin
     for I:=1 to Cel^.N_maquinas do
     begin
          if (I <> Indice)
          then begin
               if (Cel^.maquina[I].Ident_Maquina=Ident)
               then begin
                    Cel^.maquina[I].cond_operacao:=saturado;
                    end;
               end;
     end;
end;

procedure desatura_outras_maquinas (Cel:ptr_celula;Indice, Ident:byte);
var
   I:byte;
begin
     for I:=1 to Cel^.N_maquinas do
     begin
          if (I <> Indice)
          then begin
               if (Cel^.maquina[I].Ident_Maquina=Ident)
               then begin
                    Cel^.maquina[I].cond_operacao:=normal;
                    end;
               end;
     end;
end;

procedure gera_reparo_maq(Maquina:byte;Num_Cel,Indice_Maq:byte; TN:ptr_celula);
var
   Reparo:ptr_evento;
   T_reparo:real;
begin
     GetMem(Reparo,SizeOf(registro_evento));
     Reparo^.operador:=reparo_maquina;
     Reparo^.identifica_maquina:=Maquina;
     Reparo^.Ident_Num_Cel:=Num_Cel;
     Reparo^.Indice_Mac:=Indice_maq;
     tempo_reparo(TN,Indice_maq,T_reparo);
     Reparo^.T_ocorrencia:=relogio+T_reparo;
     T_reparo_maq[Num_cel,Indice_Maq]:=T_reparo_maq[Num_cel,Indice_Maq]+T_reparo;
     insere_evento(raiz,Reparo);
end;

procedure quebra_de_maquina;
var
   T2,Temp,Aux:ponteiro;
   Pont:ptr_evento;
   K_Celula:ptr_celula;
   IndiceMaq,Numero,Num_Cel,Machine,chave:byte;
   M,C,I,N,TT,KK,XX:byte;
   Nao_Existe,Deleta,Indice2:boolean;
   Cel_2:ptr_celula;
   Subst:real;
begin
     Nao_Existe:=true;
     Num_Cel:=evento^.Cell;
     chave:=evento^.Maquina;
     encontra_so_celula(Num_Cel,Raiz_Celula,K_Celula);
     IndiceMaq:= evento^.Indice;
     K_Celula^.maquina[IndiceMaq].cond_operacao:=pane;
     if (K_Celula^.maquina[IndiceMaq].situacao=ocupado)
     then begin
          encontra_evento(IndiceMaq,K_Celula^.Ident_celula,raiz,Pont);
          if Pont <> nil
          then begin
               deleta_evento(raiz,Pont);
               FreeMem(Pont,SizeOf(registro_evento));
               end;
          K_celula^.maquina[IndiceMaq].situacao:=desocupado;
          Aux:=Work[chave,IndiceMaq];
          Work[chave,IndiceMaq]:=nil;
          maquina_a_ser_ocupada(Aux,TT,KK,XX,Cel_2,Deleta);
          Aux^.T_ordenacao:=relogio;
          ocupa_manipulador(Num_Cel,Aux,K_Celula,IndiceMaq,chave);
```

```
          end
     else begin
          if (K_Celula^.manip.situacao=ocup) and (Manip[Num_Cel]^.operador=chave)
          then begin
               encontra_evento_manip(Num_Cel,IndiceMaq,raiz,Pont);
               if Pont <> nil
               then begin
                    deleta_evento(raiz,Pont);
                    FreeMem(Pont,SizeOf(registro_evento));
                    end;
               K_Celula^.manip.situacao:=nao_ocupado;
               Aux:=Manip[Num_Cel];
               Manip[Num_Cel]:=nil;
               maquina_a_ser_ocupada(Aux,TT,KK,XX,Cel_2,Deleta);
               Aux^.T_ordenacao:=relogio;
               ocupa_manipulador(Num_cel,Aux,K_celula,IndiceMaq,chave);
               end;
          end;
     Descarrega_ST(chave);
     if N_fila_maq[chave] <> 0
     then begin
          numero_maquinas_cel(chave,K_Celula,Numero);
          if Numero = 1
          then begin
               Temp:=Pont_Celula[Num_Cel];
               while (Temp <> nil) and (Nao_Existe) do
               begin
                    encontra_maq_lista_celula(chave,Temp,T2);
                    if T2=nil
                    then begin
                         Nao_Existe:=false;
                         end
                    else begin
                         Temp:=T2;
                         maquina_a_ser_ocupada(T2,M,C,I,Cel_2,Deleta);
                         if Deleta=true
                         then begin
                              Decrementa_fila_maquina(chave);
                              if Condicao_Saturacao[chave] <> 0
                              then Estado_Saturacao[chave]:=Estado_Saturacao[chave]-1;
                              Maq_na_celula(T2^.operador,K_Celula,Indice2);
                              if Indice2=true
                              then begin
                                   N:=T2^.operador;
                                   Incrementa_fila_maquina(N);
                                   satura(N,K_Celula);
                                   end;
                              end;
                         end;
               end;
               end
          else begin
               if N_fila_maq[chave] <> 0
               then begin
                    Condicao_Saturacao[chave]:=Condicao_Saturacao[chave]-K_celula^.maquina[IndiceMaq].cond_saturacao;
                    if (Estado_Saturacao[chave] < Condicao_Saturacao[chave])
                    then begin
                         satura_outras_maquinas(K_Celula,IndiceMaq,chave);
                         end;
                    end;
               end;
          gera_reparo_maq(chave,Num_Cel,IndiceMaq,K_Celula);
          end;
end;

procedure reparo_de_maquina;
var
   Ind_Maquina,N_Cel, Ident_machine, Numero:byte;
   Cell:ptr_celula;
   Ptr:ptr_evento;
   Temp:ponteiro;
begin
     Ident_machine:=evento^.Identifica_maquina;
     Ind_maquina:=evento^.Indice_Mac;
     N_Cel:=evento^.Ident_Num_Cel;
     encontra_so_celula(N_Cel,raiz_celula,Cell);
     if (Condicao_Saturacao[Ident_machine] <> 0)
     then begin
          numero_maquinas_cel (Ident_machine,Cell,Numero);
          if Numero > 1
```

```
          then begin
               Condicao_Saturacao[Ident_machine]:=Condicao_Saturacao[Ident_machine]+Cell^.maquina[Ind_maquina].Cond_Saturacao;
               if (Estado_Saturacao[Ident_machine] ( Condicao_Saturacao[Ident_machine])
               then begin
                    desatura_outras_maquinas(Cell,Ident_machine,Ind_maquina);
                    Cell^.maquina[Ind_maquina].cond_operacao:=normal;
                    end
               else begin
                    Cell^.maquina[Ind_maquina].cond_operacao:=saturado;
                    end;
               end
          else begin
               if (Estado_Saturacao[Ident_machine] ( Condicao_Saturacao[Ident_machine])
               then Cell^.maquina[Ind_maquina].cond_operacao:=normal
               else Cell^.maquina[Ind_maquina].cond_operacao:=saturado;
               end;
          end
     else Cell^.maquina[Ind_maquina].cond_operacao:=normal;
     if (Cell^.manip.situacao=nao_ocupado) and (N_fila_manipulador[N_Cel] () 0)
     then begin
          encontra_maq_lista_celula(Ident_machine,Pont_celula[N_Cel],Temp);
          if Temp () nil
          then begin
               delecao(Pont_Celula[N_Cel],Temp);
               if Temp^.maq() 0
               then begin
                    if (Temp=Output_Maq[N_Cel,Temp^.maq]) then Output_Maq[N_Cel,Temp^.maq]:=nil;
                    end;
               Manip[N_Cel]:=Temp;
               GetMem(Ptr,SizeOf(registro_evento));
               Ptr^.operador:=manipulador;
               Ptr^.Ident_celula:=N_cel;
               Ptr^.ident_atividade:=para_maquina;
               if (Cell^.manip.veloc () 0)
               then Ptr^.T_ocorrencia:=relogio+Dist_InCelula[N_Cel,Temp^.maq,Ind_maquina]/Cell^.manip.veloc
               else Ptr^.T_ocorrencia:=relogio;
               T_ocupado_manip[N_cel]:=T_ocupado_manip[N_Cel]+Ptr^.T_ocorrencia-relogio;
               T_util_manip[N_cel]:=T_util_manip[N_Cel]+Ptr^.T_ocorrencia-relogio;
               Ptr^.Indice_machine:=Ind_Maquina;
               insere_evento(raiz,Ptr);
               decrementa_fila_manipulador(N_Cel);
               decrementa_fila_maquina(Ident_machine);
               Incrementa_espera(Temp);
               end;
          end;
     gera_quebra_maquina(Ident_machine,Ind_Maquina,Cell);
end;

end.(Fim da Unit)

Unit Lista_celula;(Manipula com lista de celulas)
Interface
Uses Dos, Crt, Defvar;

     procedure del_prim_celula(prim:ptr_celula;var TN:ptr_celula);
     procedure insere_prim_celula(prim:ptr_celula;Tn:ptr_celula);
     procedure insere_celula(prim:ptr_celula;TN:ptr_celula);
     procedure deleta_celula(prim:ptr_celula;var TN:ptr_celula);
     procedure mostra_lista_celula(prim:ptr_celula);
     procedure encontra_so_celula(chave:byte; prim:ptr_celula;var Cell:ptr_celula);
     procedure encontra_maquina_celula(chave:byte;prim:ptr_celula;var Cell:ptr_celula;var Indice_maq:byte;var Acabou:boolean);
     procedure encontra_celula(chave:byte; prim:ptr_celula; var Cell:ptr_celula; var Indice_Maq:byte);
     procedure numero_maquinas_cel(chave:byte;TN:ptr_celula;var Number:byte);
     procedure encontra_maquina(chave:byte;TN:ptr_celula; var Indice_Maq:byte);
     procedure identifica(chave:palavra;prim:ptr_celula; var Cell:ptr_celula;var Indice_maq:byte; var K:byte);
     procedure satura(maquina:byte;Cell:ptr_celula);
     procedure desatura(machine:byte;Cell:ptr_celula);
     procedure saturacao_maquina(chave:byte;TN:ptr_celula);
     procedure desaturacao_maquina(chave:byte;TN:ptr_celula);
     procedure deleta_maquina(chave:palavra;prim:ptr_celula; var Cell:ptr_celula);
     procedure Maq_na_celula(chave:byte;K_Cel:ptr_celula;var Maq_Celula:boolean);

Implementation

procedure del_prim_celula(prim:ptr_celula;var TN:ptr_celula);
begin
     TN:=prim^.ptr_proximo;
     if TN^.ptr_proximo=nil
     then begin
          prim^.ptr_proximo:=nil
```

```
          end
     else begin
          prim^.ptr_proximo:=TN^.ptr_proximo;
          end
end;

procedure insere_prim_celula(prim:ptr_celula;TN:ptr_celula);
begin
     if prim^.ptr_proximo=nil
     then begin
          TN^.ptr_proximo:=nil;
          prim^.ptr_proximo:=TN;
          end
     else begin
          TN^.ptr_proximo:=prim^.ptr_proximo;
          prim^.ptr_proximo:=TN;
          end;
end;

procedure insere_celula(prim:ptr_celula;TN:ptr_celula);
var
   posicao:ptr_celula;
   KK:ptr_celula;
   Inseri:boolean;
begin
     Inseri:=true;
     KK:=prim;
     posicao:=prim^.ptr_proximo;
     while Inseri do
     begin
         if (posicao=nil)
         then begin
              TN^.ptr_proximo:=posicao;
              KK^.ptr_proximo:=TN;
              Inseri:=false;
              end
         else begin
              if (posicao^.Ident_celula ) TN^.Ident_celula)
              then begin
                   TN^.ptr_proximo:=posicao;
                   KK^.ptr_proximo:= TN;
                   Inseri:=false;
                   end;
              end;
         KK:=posicao;
         posicao:=posicao^.ptr_proximo;
     end;
end;

procedure deleta_celula(prim:ptr_celula;var TN:ptr_celula);
var
   Prov:ptr_celula;
   Deletei:boolean;
begin
     Prov:=prim;
     Deletei:=true;
     while Deletei and (Prov () nil) do
     begin
          if Prov^.ptr_proximo = TN
          then begin
               if TN^.ptr_proximo=nil
               then begin
                    Prov^.ptr_proximo:=nil;
                    Deletei:=false;
                    end
               else begin
                    Prov^.ptr_proximo:=TN^.ptr_proximo;
                    Deletei:=false;
                    end;
               end;
          Prov:=Prov^.ptr_proximo;
          end;
end;

procedure mostra_lista_celula(prim:ptr_celula); (* rotina de depuracao *)
var
   posicao:ptr_celula;
   proximo:char;
begin
     posicao:=prim^.ptr_proximo;
```

```
     while posicao <> nil do
     begin
          writeln('Ident_celula',posicao^.Ident_celula);
          posicao:=posicao^.ptr_proximo;
          if posicao = nil
          then writeln('aponta para o fim')
          else writeln('aponta para ',posicao^.Ident_celula);
end; end;

procedure encontra_so_celula (chave:byte; prim:ptr_celula; var Cell:ptr_celula);
var
   Achei:boolean;
   TN:ptr_celula;
begin
     TN:=prim^.ptr_proximo;
     Achei:=true;
     while Achei and (TN<>nil) do
     begin
          if TN^.Ident_celula=chave
          then begin
               Achei:=false;
               Cell:=TN;
               end
          else begin
               TN:=TN^.ptr_proximo;
               end;
     end;
end;

procedure encontra_maquina_celula(chave:byte; prim:ptr_celula; var Cell:ptr_celula; var Indice_Maq:byte;var Acabou:boolean);
var
   Nao_achou:boolean;
   TN:ptr_celula;
   I:byte;
begin
     Indice_maq:=0;
     TN:=prim^.ptr_proximo;
     Nao_achou:=true;
     Acabou:=false;
     while Nao_achou and (TN<>nil) do
     begin
          I:=1;
          repeat
               if (TN^.maquina[I].Ident_maquina=chave) and (TN^.maquina[I].cond_operacao=normal)
               then begin
                    Nao_achou:=false;
                    Acabou:=true;
                    Indice_Maq:=I;
                    Cell:=TN;
                    end;
                I:=I+1;
                until (I > TN^.N_maquinas) or Acabou;
           TN:=TN^.ptr_proximo;
     end;
end;

procedure encontra_celula(chave:byte; prim:ptr_celula; var Cell:ptr_celula; var Indice_Maq:byte);
var
   Nao_achou:boolean;
   Acabou:boolean;
   TN:ptr_celula;
   I:byte;
begin
     Indice_maq:=0;
     TN:=prim^.ptr_proximo;
     Nao_achou:=true;
     Acabou:=false;
     Existe:=false;
     while Nao_achou and (TN<>nil) do
     begin
          I:=1;
          repeat
               if (TN^.maquina[I].Ident_maquina=chave)
               then begin
                    if (TN^.maquina[I].situacao=desocupado) and (TN^.maquina[I].cond_operacao=normal)
                    then begin
                         Nao_achou:=false;
                         Acabou:=true;
                         Indice_Maq:=I;
                         end;
```

```
                    Cell:=TN;
                    Existe:=true;
                    end;
                I:=I+1;
                until (I ) TN^.N_maquinas) or Acabou;
            TN:=TN^.ptr_proximo;
        end;
        if Existe=false
        then begin
             clrscr;
             writeln ('ERRO NA ENTRADA DE DADOS!!!!!!!!!!!!');
             writeln ;
             writeln ('TECLE (CTR) (BREAK) PARA INTERROMPER EXECUCAO');
             writeln;
             writeln ('VERIFIQUE ARQUIVO DE ENTRADA, POR FAVOR');
             readln;
        end;
end;

procedure numero_maquinas_cel (chave:byte;TN:ptr_celula;var Number:byte);
var
   I:byte;
begin
     I:=1;
     Number:=0;
     repeat
          if (TN^.maquina[I].Ident_maquina=chave)
          then begin
               Number:=Number+1;
               end;
          I:=I+1;
          until (I)TN^.N_maquinas);
end;

procedure encontra_maquina (chave:byte;TN:ptr_celula;var Indice_Maq:byte);
var
   Nao_achou:boolean;
   I:byte;
begin
     Indice_maq:=0;
     Nao_achou:=false;
     Existe:=false;
     I:=1;
     repeat
          if (TN^.maquina[I].Ident_maquina=chave)
          then begin
               if (TN^.maquina[I].situacao=desocupado) and (not(TN^.maquina[I].cond_operacao=pane))
               then begin
                    Nao_achou:=true;
                    Indice_Maq:=I;
                    end;
               Existe:=true;
               end;
          I:=I+1;
          until (I)TN^.N_maquinas) or Nao_achou;
end;

procedure identifica(chave:palavra; prim:ptr_celula; var Cell:ptr_celula; var Indice_maq:byte; var K:byte);
var
   Nao_achou:boolean;
   Acabou:boolean;
   TN:ptr_celula;
   I:byte;
begin
     TN:=prim^.ptr_proximo;
     Nao_achou:=true;
     Acabou:=false;
     while (TN()nil) and Nao_achou do
     begin
          I:=1;
          repeat
               if TN^.maquina[I].name=chave
               then begin
                    Nao_achou:=false;
                    Acabou:=true;
                    Indice_Maq:=I;
                    K:=TN^.maquina[I].Ident_maquina;
                    Cell:=TN;
                    end;
               I:=I+1;
```

```
                until (I ) TN^.N_maquinas) or Acabou;
            TN:=TN^.ptr_proximo;
        end;
end;

procedure satura(Maquina:byte;Cell:ptr_celula);
begin
     if (Condicao_Saturacao[Maquina] () 0)
          then begin
               Estado_Saturacao[Maquina]:=Estado_Saturacao[Maquina]+1;
               if (Estado_Saturacao[Maquina])=Condicao_Saturacao[Maquina])
               then begin
                    saturacao_maquina(Maquina,Cell);
                    end;
               end;
end;

procedure desatura(Machine:byte;Cell:ptr_celula);
begin
     if (Condicao_Saturacao[Machine] () 0)
     then begin
          Estado_Saturacao[Machine]:=Estado_Saturacao[Machine]-1;
          if (Estado_Saturacao[Machine] ( Condicao_Saturacao[Machine])
          then begin
               desaturacao_maquina(Machine,Cell);
               end;
          end;
end;

procedure saturacao_maquina(chave:byte; TN:ptr_celula);
var
   I:byte;
begin
     for I:=1 to TN^.N_maquinas do
     begin
          if (TN^.maquina[I].Ident_maquina=chave)
          then begin
               TN^.maquina[I].cond_operacao:=saturado;
               end;
     end;
end;

procedure desaturacao_maquina(chave:byte; TN:ptr_celula);
var
   I:byte;
begin
     for I:=1 to TN^.N_maquinas do
     begin
          if (TN^.maquina[I].Ident_maquina=chave)
          then begin
               TN^.maquina[I].cond_operacao:=normal;
               end;
     end;
end;

procedure deleta_maquina(chave:palavra;prim:ptr_celula; var Cell:ptr_celula);
var
   K, I, J:byte;
begin
     identifica (chave, prim, Cell, I, J);
     if Cell^.N_maquinas = 1
          then deleta_celula(prim,Cell)
          else begin
               if I=Cell^.N_maquinas then Cell^.N_maquinas:=Cell^.N_maquinas-1
               else begin
                    for K:=I to (Cell^.N_maquinas-1) do
                    begin
                        Cell^.maquina[K]:=Cell^.maquina[K+1];
                    end;
                    Cell^.N_maquinas:=Cell^.N_maquinas-1;
                    end;
               end;
end;

procedure Maq_na_celula(chave:byte;K_Cel:ptr_celula;var Maq_Celula:boolean);
var
   T:byte;
begin
     Maq_celula:=false;
     for T:=1 to K_Cel^.N_maquinas do
```

```
        begin
            if (K_Cel^.maquina[T].Ident_maquina=chave)
            then Maq_celula:=true;
        end;
    end;
    end.(Fim da Unit)

    Unit Lista_tarefa;(Manipula com lista de tarefas)
    Interface
    Uses Dos, Defvar;

        procedure determina_ordenador(Prioridade:byte;TN:ponteiro);
        procedure del_prim_el(prim:ponteiro; var TN:ponteiro);
        procedure insere_prim_el(prim:ponteiro; TN:ponteiro);
        procedure cria_lista_ordenada(prim:ponteiro; TN:ponteiro);
        procedure insere_prioridade(prim:ponteiro; TN:ponteiro);
        procedure mostra_lista_tarefa(prim:ponteiro);
        procedure acha_elemento(chave:byte;prim:ponteiro;var TN:ponteiro);
        procedure delecao(prim:ponteiro; var TN:ponteiro);

    Implementation

    procedure determina_ordenador(Prioridade:byte;TN:ponteiro);
    begin
        case Prioridade of
        1:TN^.ordenador:=TN^.T_ordenacao;
        2:TN^.ordenador:=TN^.prioridade;
        3:TN^.ordenador:=TN^.T_entrega_final;
        4:TN^.ordenador:=TN^.N_op_completar;
        end;
    end;

    procedure del_prim_el(prim:ponteiro;var TN:ponteiro);
    begin
        TN:=prim^.ptr_proximo;
        if TN^.ptr_proximo=nil
        then begin
            prim^.ptr_proximo:=nil
            end
        else begin
            prim^.ptr_proximo:=TN^.ptr_proximo;
            end
    end;

    procedure insere_prim_el(prim:ponteiro;TN:ponteiro);
    begin
        if prim^.ptr_proximo=nil
        then begin
            TN^.ptr_proximo:=nil;
            prim^.ptr_proximo:=TN;
            end
        else begin
            TN^.ptr_proximo:=prim^.ptr_proximo;
            prim^.ptr_proximo:=TN;
            end;
    end;

    procedure cria_lista_ordenada(prim:ponteiro;TN:ponteiro);
    var
       posicao:ponteiro;
       KK:ponteiro;
       Inseri:boolean;
    begin
        Inseri:=true;
        KK:=prim;
        posicao:=prim^.ptr_proximo;
        while Inseri do
        begin
            if (posicao=nil)
            then begin
                TN^.ptr_proximo:=posicao;
                KK^.ptr_proximo:=TN;
                Inseri:=false;
                end
            else begin
                if (posicao^.T_ordenacao ) TN^.T_ordenacao)
                then begin
                    TN^.ptr_proximo:=posicao;
                    KK^.ptr_proximo:= TN;
                    Inseri:=false;
```

```
                end;
            end;
        KK:=posicao;
        posicao:=posicao^.ptr_proximo;
    end;
end;

procedure insere_prioridade(prim:ponteiro;TN:ponteiro);
var
   posicao:ponteiro;
   KK:ponteiro;
   Inseri:boolean;
begin
     determina_ordenador(Sist_prioridade,TN);
     Inseri:=true;
     KK:=prim;
     posicao:=prim^.ptr_proximo;
     while Inseri do
     begin
         if (posicao=nil)
         then begin
              TN^.ptr_proximo:=posicao;
              KK^.ptr_proximo:=TN;
              Inseri:=false;
              end
         else begin
              if (posicao^.ordenador ) TN^.ordenador)
              then begin
                   TN^.ptr_proximo:=posicao;
                   KK^.ptr_proximo:= TN;
                   Inseri:=false;
                   end;
              end;
         KK:=posicao;
         posicao:=posicao^.ptr_proximo;
    end;
end;

procedure mostra_lista_tarefa(prim:ponteiro); (* rotina de depuracao *)
var
   posicao:ponteiro;
   proximo:char;
begin
     posicao:=prim^.ptr_proximo;
     if posicao=nil then writeln(' Lista vazia');
     while posicao () nil do
     begin
          writeln ('Tempo de ordenacao=',posicao^.T_ordenacao);
          writeln('Operador= ',posicao^.operador);
          writeln;
          posicao:=posicao^.ptr_proximo;
          if posicao = nil
          then writeln('aponta para o fim');
     end;
end;

procedure acha_elemento(chave:byte;prim:ponteiro;var TN:ponteiro);
var
   Nao_achou:boolean;
begin
     TN:=prim^.ptr_proximo;
     Nao_achou:=true;
     while (TN()nil) and Nao_achou do
     begin
         if TN^.identificacao=chave
         then Nao_achou:=false
         else TN:=TN^.ptr_proximo;
     end;
end;

procedure delecao(prim:ponteiro;var TN:ponteiro);
var
   Prov:ponteiro;
   Deletei:boolean;
begin
     Prov:=prim;
     Deletei:=true;
     while Deletei and (Prov () nil) do
     begin
          if Prov^.ptr_proximo = TN
```

```
          then begin
               if TN^.ptr_proximo=nil
               then begin
                    Prov^.ptr_proximo:=nil;
                    Deletei:=false;
                    end
               else begin
                    Prov^.ptr_proximo:=TN^.ptr_proximo;
                    Deletei:=false;
                    end;
               end;
          Prov:=Prov^.ptr_proximo;
          end;
end;
end.(fim da unit)

Unit lista_evento;(Manipula com lista de eventos)
Interface
Uses Dos, Defvar;

     procedure del_prim_evento(prim:ptr_evento;var TN:ptr_evento);
     procedure insere_prim_evento(prim:ptr_evento;TN:ptr_evento);
     procedure insere_evento(prim:ptr_evento;TN:ptr_evento);
     procedure deleta_evento(prim:ptr_evento;var TN:ptr_evento);
     procedure mostra_lista_evento(prim:ptr_evento);
     procedure encontra_evento(chave1,chave2:byte;prim:ptr_evento;var Retorno:ptr_evento);
     procedure encontra_evento_manip(chave1,chave2:byte;prim:ptr_evento;var Retorno:ptr_evento);
     procedure encontra_evento_ST(chave1:byte;prim:ptr_evento;var Retorno:ptr_evento);

Implementation

procedure del_prim_evento(prim:ptr_evento;var TN:ptr_evento);
begin
     TN:=prim^.ptr_proximo;
     if TN^.ptr_proximo=nil
     then begin
          prim^.ptr_proximo:=nil
          end
     else begin
          prim^.ptr_proximo:=TN^.ptr_proximo;
          end
end;

procedure insere_prim_evento(prim:ptr_evento;TN:ptr_evento);
begin
     if prim^.ptr_proximo=nil
     then begin
          TN^.ptr_proximo:=nil;
          prim^.ptr_proximo:=TN;
          end
     else begin
          TN^.ptr_proximo:=prim^.ptr_proximo;
          prim^.ptr_proximo:=TN;
          end;
end;

procedure insere_evento(prim:ptr_evento;TN:ptr_evento);
var
   posicao:ptr_evento;
   KK:ptr_evento;
   Inseri:boolean;
begin
     Inseri:=true;
     KK:=prim;
     posicao:=prim^.ptr_proximo;
     while Inseri do
     begin
        if (posicao=nil)
        then begin
             TN^.ptr_proximo:=posicao;
             KK^.ptr_proximo:=TN;
             Inseri:=false;
             end
        else begin
             if (posicao^.T_ocorrencia) TN^.T_ocorrencia)
             then begin
                  TN^.ptr_proximo:=posicao;
                  KK^.ptr_proximo:= TN;
                  Inseri:=false;
                  end;
```

```
            end;
        KK:=posicao;
        posicao:=posicao^.ptr_proximo;
    end;
end;

procedure deleta_evento(prim:ptr_evento;var TN:ptr_evento);
var
   Prov:ptr_evento;
   Deletei:boolean;
begin
    Prov:=prim;
    Deletei:=true;
    while Deletei and (Prov () nil) do
    begin
        if Prov^.ptr_proximo = TN
        then begin
            if TN^.ptr_proximo=nil
            then begin
                Prov^.ptr_proximo:=nil;
                Deletei:=false;
                end
            else begin
                Prov^.ptr_proximo:=TN^.ptr_proximo;
                Deletei:=false;
                end;
            end;
        Prov:=Prov^.ptr_proximo;
        end;
end;

procedure mostra_lista_evento(prim:ptr_evento); (* rotina de depuracao *)
var
   posicao:ptr_evento;
   proximo:char;
begin
    posicao:=prim^.ptr_proximo;
    if posicao=nil then writeln('Lista vazia');
    while posicao () nil do
    begin
        writeln('Tempo de ocorrencia = ',posicao^.T_ocorrencia);
        if (posicao^.operador=maquina)
        then begin
            writeln('Maquina = ',posicao^.Ident_maquina);
            writeln('celula = ',posicao^.Ident_celula);
            end;
        if posicao^.operador=manipulador
        then begin
            (writeln('manipulador = ',posicao^.Ident_atividade);)
            writeln(' Celula = ', posicao^.Ident_celula);
            end;
        posicao:=posicao^.ptr_proximo;
        if posicao = nil
        then writeln('aponta para o fim')
        (else writeln('aponta para ',posicao^.Identificacao));
    end;
end;

procedure encontra_evento (chave1:byte; chave2:byte;prim:ptr_evento; var Retorno:ptr_evento);
var
   Temp:ptr_evento;
   Nao_Achei:boolean;
begin
    Retorno:=nil;
    Temp:=prim^.ptr_proximo;
    Nao_Achei:=true;
    while (Temp () nil) and (Nao_Achei) do
    begin
        if Temp^.operador=maquina
        then begin
            if (Temp^.Ident_Cell=chave2) and (Temp^.Indice_Maq=chave1)
            then begin
                Nao_Achei:=false;
                Retorno:=Temp;
                end;
            end;
        Temp:=Temp^.ptr_proximo;
    end;
end;
```

```
procedure encontra_evento_manip(chave1,chave2:byte;prim:ptr_evento;var Retorno:ptr_evento);
var
   Temp:ptr_evento;
   Nao_Achei:boolean;
begin
     Retorno:=nil;
     Temp:=prim^.ptr_proximo;
     Nao_Achei:=true;
     while (Temp <> nil) and (Nao_Achei) do
     begin
          if Temp^.operador=manipulador
          then begin
               if (Temp^.Ident_Celula=chave1) and (Temp^.Indice_Machine=chave2)
               then begin
                    Nao_Achei:=false;
                    Retorno:=Temp;
                    end;
               end;
          Temp:=Temp^.ptr_proximo;
     end;
end;

procedure encontra_evento_ST(chave1:byte;prim:ptr_evento;var Retorno:ptr_evento);
var
   Temp:ptr_evento;
   Nao_Achei:boolean;
begin
     Retorno:=nil;
     Temp:=prim^.ptr_proximo;
     Nao_Achei:=true;
     while (Temp <> nil) and (Nao_Achei) do
     begin
          if Temp^.operador=SisTrans
          then begin
               if (Temp^.Ident_ST=chave1)
               then begin
                    Nao_Achei:=false;
                    Retorno:=Temp;
                    end;
               end;
          Temp:=Temp^.ptr_proximo;
     end;
end;

end.(Final da Unit)
{$O+,F+}
Unit dialogo;
Interface
uses dos, overlay, crt, dist, varler, defent1, defent2, Tela, Funcoes2, Le_funcoes, letecla;
{$O Tela}
{$O Funcoes2}
{$O Le_funcoes}
{$O dist}
{$O defent1}
{$O defent2}
    procedure entrada;
Implementation

procedure entrada;
     begin
     informacoes_iniciais;  {** le o nome do arquivo, nome da simulacao e numero de celulas **}
     ASSIGN(VardeArquivo,Nome_arquivo);
     REWRITE(VardeArquivo);
     WRITE(VardeArquivo,Nome_da_simulacao+'*');WRITELN(VardeArquivo,' (- Nome da Simulacao');
     WRITE(VardeArquivo,N_celulas);        WRITELN(VardeArquivo,' (- Numero de celulas');
     for i:=1 to N_celulas do
     begin
          le_ident_e_nmaq_de_celula(i); {** le identificacao e numero de maquinas da celula **}
          WRITE(VardeArquivo,i);                    WRITELN(VardeArquivo,' (- Identificacao da celula ',i);
          WRITE(VardeArquivo,Numero_maquinas[i]);   WRITELN(VardeArquivo,' (- Numero de maquinas da celula ',i);
          for ii:=1 to Numero_maquinas[i] DO
          begin
               le_caract_de_maquina; {** LE NOME, TEMPO DE SETUP, ETC., DE MAQUINA **}
               WRITE(VardeArquivo,Tempo_setup);  WRITELN(VardeArquivo,' (- Tempo de setup da maquina ',ii,' da celula ',i);
               WRITE(VardeArquivo,Nome_maquina+'*'); WRITELN(VardeArquivo,' (- Nome da maquina ',ii,' da celula ',i);
               WRITE(VardeArquivo,Ident_maquina);WRITELN(VardeArquivo,' (- Identificacao da maquina ',ii,' da celula ',i);
               WRITE(VardeArquivo,Cond_saturacao);WRITELN(VardeArquivo,' (- Condicao de saturacao da maquina ',i);
               le_func_prob_quebra; {** LE A FUNCAO DE PROBABILIDADE DE QUEBRA DE MAQUINA **}
               WRITE(VardeArquivo,F_prob_quebra);
```

```
          WRITELN(VardeArquivo,' (- Funcao de probabilidade de quebra da maquina ',ii,' da celula ',i);
        WRITE(VardeArquivo,Parametro1);
          WRITELN(VardeArquivo,' (- Parametro 1 da Func. prob. de quebra da maquina ',ii);
        if N_parametros)1 then begin
        WRITE(VardeArquivo,Parametro2);
          WRITELN(VardeArquivo,' (- Parametro 2 da Func. prob. de quebra da maquina ',ii);
        end;
        if N_parametros)2 then begin
        WRITE(VardeArquivo,Parametro3);
        WRITELN(VardeArquivo,' (- Parametro 3 da Func. prob. de quebra da maquina ',ii);
        end;
        le_func_prob_reparo; (** LE A FUNCAO DE PROBABILIDADE DE REPARO DE MAQUINA **)
       WRITE(VardeArquivo,F_prob_reparo);
           WRITELN(VardeArquivo,' (- Funcao de probabilidade de reparo da maquina ',ii,' da celula ',i);
       WRITE(VardeArquivo,Parametro1);
           WRITELN(VardeArquivo,' (- Parametro 1 da Func. prob. de reparo da maquina ',ii);
       if N_parametros)1 then begin
           WRITE(VardeArquivo,Parametro2);
              WRITELN(VardeArquivo,' (- Parametro 2 da Func. prob. de reparo da maquina ',ii);
       end;
       if N_parametros)2 then begin
           WRITE(VardeArquivo,Parametro3);
              WRITELN(VardeArquivo,' (- Parametro 3 da Func. prob. de reparo da maquina ',ii);
       end;
    end;
    window(3,4,75,23);
    clrscr;
    window(1,1,80,25);
    le_manipulador; (** LEITURA DE NOME E VELOCIDADE DO MANIPULADOR **)
    WRITE(VardeArquivo,Nome_manipulador+'*');      WRITELN(VardeArquivo,' (- Nome do manipulador da celula ',i);
    WRITE(VardeArquivo,Velocidade_manipulador);WRITELN(VardeArquivo,' (- Velocidade do manipulador da celula ',i);
end;
  distancias;
  le_numero_componentes; (*** LEITURA DO NUMERO DE COMPONENTES ***)
  WRITE(VardeArquivo,N_componentes); WRITELN(VardeArquivo,' (- Numero de componentes');
  for i2:=1 to N_componentes DO
  begin
    resposta:='n';
    while (resposta='n') or (resposta='N') do
    begin
        le_nome_componente; (** LE O NOME DO COMPONENTE **)
        Ident_componente:=i2; (** LE A IDENTIFICACAO DO COMPONENTE **)
        le_F_prob_T_cheg; (*** LE A FUNCAO DE PROBABILIDADE DE TEMPO DE CHEGADA DO COMPONENTE ***)
    end;
    le_parametros_T_cheg; (*** LE OS PARAMETROS DA F. PROB. TEMPO DE CHEGADA DO COMPONENTE ***)
    WRITE(VardeArquivo,Nome_componente+'*');
     WRITELN(VardeArquivo,' (- Nome do componente ',i2);
    WRITE(VardeArquivo,Ident_componente);
     WRITELN(VardeArquivo,' (- Identificacao do componente ',i2);
    WRITE(VardeArquivo,F_prob_tempo_chegada);
     WRITELN(VardeArquivo,' (- Funcao prob. tempo chegada do componente ',i2);
    WRITE(VardeArquivo,Parametro1);
     WRITELN(VardeArquivo,' (- Parametro 1 da Func. prob. tempo chegada do componente ',i2);
    if N_parametros)1 then begin
     WRITE(VardeArquivo,Parametro2);
     WRITELN(VardeArquivo,' (- Parametro 2 da Func. prob. tempo chegada do componente ',i2);
      end;
    if N_parametros)2 then begin
     WRITE(VardeArquivo,Parametro3);
     WRITELN(VardeArquivo,' (- Parametro 3 da Func. prob. tempo chegada do componente ',i2);
      end;
    le_F_Prob_Tam_lote; (*** LE A FUNCAO DE PROBABILIDADE DE TAMANHO DE LOTE DO COMPONENTE ***)
    le_parametros_Tam_lote; (*** LE OS PARAMETROS DA F. PROB. TAMANHO DE LOTE DO COMPONENTE ***)
    WRITE(VardeArquivo,F_prob_tamanho_lote);
      WRITELN(VardeArquivo,' (- Funcao prob. tamanho de lote do componente ',i2);
    WRITE(VardeArquivo,Parametro1);
      WRITELN(VardeArquivo,' (- Parametro 1 da Func. prob. tamanho de lote do componente ',i2);
    if N_parametros)1 then
      begin
       WRITE(VardeArquivo,Parametro2);
       WRITELN(VardeArquivo,' (- Parametro 2 da Func. prob. tamanho de lote do componente ',i2);
      end;
    if N_parametros)2 then
      begin
       WRITE(VardeArquivo,Parametro3);
       WRITELN(VardeArquivo,' (- Parametro 3 da Func. prob. tamanho de lote do componente ',i2);
      end;
    le_prioridade_componente;
    WRITE(VardeArquivo,prioridade_componente);
    WRITELN(VardeArquivo,' (- Prioridade do Componente ',i2);
```

```
          le_numero_operacoes; {** LE O NUMERO DE OPERACOES **)
          WRITE(VardeArquivo,N_operacoes);
          for i3:=1 to N_operacoes do
          begin
              le_operacao; {*** LEITURA DE TODAS AS OPCOES DE MAQUINA, COM F. DE PROB. E PARAMETROS **)
              WRITE(VardeArquivo,opcao_1_maquina);
               WRITELN(VardeArquivo,' (- Opcao 1 de maquina da operacao ',i3,'do componente ',i2);
              WRITE(VardeArquivo,F_prob_opcao_1);
               WRITELN(VardeArquivo,' (- Funcao prob. opcao 1 da operacao ',i3,' do componente ',i2);
              WRITE(VardeArquivo,opcao_2_maquina);
               WRITELN(VardeArquivo,' (- Opcao 2 de maquina da operacao ',i3,' do componente ',i2);
              WRITE(VardeArquivo,F_prob_opcao_2);
               WRITELN(VardeArquivo,' (- Funcao prob. opcao 2 da operacao ',i3,' do componente ',i2);
              WRITE(VardeArquivo,opcao_3_maquina);
               WRITELN(VardeArquivo,' (- Opcao 3 de maquina da operacao ',i3,' do componente ',i2);
              WRITE(VardeArquivo,F_prob_opcao_3);
               WRITELN(VardeArquivo,' (- Funcao prob. opcao 3 da operacao ',i3,' do componente ',i2);
              WRITE(VardeArquivo,N_par1);
               WRITELN(VardeArquivo,' (- Numero de parametros da funcao da operacao ',i3,' da opcao 1 do comp. ',i2);
              WRITE(VardeArquivo,par11);
               WRITELN(VardeArquivo,' (- Parametro 1.1');
              if N_par1>1 then
                begin
                 WRITE(VardeArquivo,par12);
                 WRITELN(VardeArquivo,' (- Parametro 1.2');
                end;
              if N_par1>2 then
               begin
                WRITE(VardeArquivo,par13);
                WRITELN(VardeArquivo,' (- Parametro 1.3');
               end;
              WRITE(VardeArquivo,N_par2);
               WRITELN(VardeArquiivo,' (- Numero de parametros da funcao da operacao ',i3,' da opcao 2 do comp. ',i2);
              WRITE(VardeArquivo,par21);
               WRITELN(VardeArquivo,' (- Parametro 2.1');
              if N_par2>1 then
               begin
                WRITE(VardeArquivo,par22);
                WRITELN(VardeArquivo,' (- Parametro 2.2');
               end;
              if N_par2>2 then
               begin
                WRITE(VardeArquivo,par23);
                WRITELN(VardeArquivo,' (- Parametro 2.3');
               end;
              WRITE(VardeArquivo,N_par3);
               WRITELN(VardeArquivo,' (- Numero de parametros da funcao da operacao ',i3,' da opcao 3 do comp. ',i2);
              WRITE(VardeArquivo,par31);
               WRITELN(VardeArquivo,' (- Parametro 3.1');
              if N_par3>1 then
               begin
                WRITE(VardeArquivo,par32);
                WRITELN(VardeArquivo,' (- Parametro 3.2');
               end;
              if N_par3>2 then
               begin
                WRITE(VardeArquivo,par33);
                WRITELN(VardeArquivo,' (- Parametro 3.3');
               end;
          end;
          window(3,3,78,24);
          clrscr;
          window(1,1,80,25);
        end;
      entrada_sistema_transferencia;
      gravacao_sistema_transferencia;
      window(3,3,78,24);
      clrscr;
      window(1,1,80,25);
      gotoxy(10,12);
      writeln('TECLE QUALQUER TECLA P/ FECHAR O ARQUIVO');
      gotoxy(10,15);
      readln;
      gotoxy(10,15);
      writeln('********** FECHANDO ARQUIVO **********');
      CLOSE(VardeArquivo);
end;
end.
{$O+,F+}
Unit Tela;
```

```
Interface
Uses Dos, Overlay, Crt, letecla, Varler;
   procedure primeira_tela;
   procedure primeira_tela_leitura;
   procedure tela_prioridade;
   procedure segunda_tela(i:byte);
   procedure seg_tela_2;
   procedure seg_tela_3;
   procedure terceira_tela;
   procedure menu_principal;
   procedure moldura2(ColTopoEsq,LinhaTopo,ColInfDir,LinhaInf:integer);
   procedure moldura(ColTopoEsq,LinhaTopo,ColInfDir,LinhaInf:integer);

Implementation

procedure primeira_tela;
begin
     gotoxy(10,3);
     writeln('Drive do arquivo:                                    ');
     gotoxy(10,5);
     writeln('Nome do arquivo:                  ');
     gotoxy(10,10);
     writeln('Nome da simulacao:                              ');
     gotoxy(10,15);
     writeln('Numero de celulas:          ');
end;

procedure primeira_tela_leitura;
begin
     gotoxy(10,3);
     writeln('Drive do arquivo:                                    ');
     gotoxy(10,5);
     writeln('Nome do arquivo:                  ');
end;

procedure tela_prioridade;
begin
     gotoxy(10,5);
     writeln('SISTEMA DE PRIORIDADE DO SISTEMA');
     gotoxy(10,9);
     writeln('(1) Sistema FIFO                              ');
     gotoxy(10,10);
     writeln('(2) Prioridade entre componentes              ');
     gotoxy(10,11);
     writeln('(3) Menor Tempo de entrega final do componente ');
     gotoxy(10,12);
     writeln('(4) Menor numero de operacoes a completar     ');
     gotoxy(10,20);
     writeln('Escolha sua opcao:      ');
end;

procedure segunda_tela(i:byte);
begin
     gotoxy(5,3);
     writeln('Identificacao da celula ',i,' (numero inteiro):        ');
     gotoxy(5,4);
     writeln('Numero de maquinas:         ');
end;

procedure seg_tela_2;
begin
     gotoxy(5,7);
     writeln('Tempo de setup:                  ');
     gotoxy(5,8);
     writeln('Nome da maquina:                              ');
     gotoxy(5,9);
     writeln('Identificacao da Maquina (numero inteiro):      ');
     gotoxy(5,10);
     writeln('Condicao de saturacao (numero inteiro):         ');
end;

procedure seg_tela_3;
begin
     gotoxy(5,10);
     writeln('Nome do manipulador:                            ');
     gotoxy(5,14);
     writeln('Velocidade media de atuacao do manipulador:          ');
end;

procedure terceira_tela;
```

```
begin
     gotoxy(10,10);
     writeln('Numero de Componentes:          ');
end;

procedure menu_principal;
begin
     gotoxy(10,8);
     writeln('MENU PRINCIPAL');
     gotoxy(10,12);
     writeln('(1) MODULO DIALOGO                              ');
     gotoxy(10,14);
     writeln('(2) EXECUCAO              ');
     gotoxy(10,16);
     writeln('(3) FIM');
     gotoxy(10,21);
     writeln('Escolha sua opcao:      ');
end;

procedure moldura2(ColTopoEsq,LinhaTopo,ColInfDir,LinhaInf:integer);

Const
     (declaracao das constantes para a moldura)
     TE=#201; TD=#187; (Topos esquerdo e direito)
     IE=#200; ID=#188; (Inf. esquerdo e direito)
     H =#205; V =#186; (Horizontal e vertical)
Var
   X:integer;

begin
     LowVideo;
     Gotoxy(ColTopoEsq,LinhaTopo);
     Write(TE);
     Gotoxy(ColInfDir,LinhaTopo);
     write(TD);
     gotoxy(ColTopoEsq,LinhaInf);
     write(IE);
     gotoxy(ColInfDir,LinhaInf);
     write(ID);
     For X:=LinhaTopo+1 to LinhaInf-1 do
     begin
          gotoxy(ColTopoEsq,x);
          write(V);
          gotoxy(ColInfDir,x);
          write(V);
     end;
     For X:=ColTopoEsq+1 to ColInfDir-1 do
     begin
          gotoxy(X,LinhaTopo);
          write(H);
          gotoxy(X,LinhaInf);
          write(H);
     end;
     NormVideo;
end;

procedure moldura(ColTopoEsq,LinhaTopo,ColInfDir,LinhaInf:integer);
Const
     (declaracao das constantes para a moldura)
     TE=#218; TD=#191; (Topos esquerdo e direito)
     IE=#192; ID=#217; (Inf. esquerdo e direito)
     H =#196; V =#179; (Horizontal e vertical)
Var
   X:integer;
begin
     LowVideo;
     Gotoxy(ColTopoEsq,LinhaTopo);
     Write(TE);
     Gotoxy(ColInfDir,LinhaTopo);
     write(TD);
     gotoxy(ColTopoEsq,LinhaInf);
     write(IE);
     gotoxy(ColInfDir,LinhaInf);
     write(ID);
     For X:=LinhaTopo+1 to LinhaInf-1 do
     begin
          gotoxy(ColTopoEsq,x);
          write(V);
          gotoxy(ColInfDir,x);
          write(V);
```

```
     end;
     For X:=ColTopoEsq+1 to ColInfDir-1 do
     begin
          gotoxy(X,LinhaTopo);
          write(H);
          gotoxy(X,LinhaInf);
          write(H);
     end;
     NormVideo;
end;
end.(fim da Unit)

{$O+,F+}
Unit Funcoes2;
Interface
Uses Overlay, Dos, Crt, Varler, Letecla;

     procedure funcoes;
     procedure limpa_funcoes;
     procedure imprime_parametros(F_prob,coluna,linha:byte;var N_parametros:byte);
     procedure apaga_parametros;
     procedure imprime_opcoes;
     procedure pede_parametros;
Implementation

procedure funcoes;
begin
     gotoxy(6,19);
     writeln('Constante          (0)     Distr. binomial negativa (5) ');
     gotoxy(6,20);
     writeln('Distr. uniforme    (1)     Distr. binomial          (6) ');
     gotoxy(6,21);
     writeln('Distr. exponencial (2)     Distr. hipergeometrica   (7) ');
     gotoxy(6,22);
     writeln('Distr. gama        (3)     Distr. poisson           (8) ');
     gotoxy(6,23);
     writeln('Distr. normal      (4)     Distr. m-Erlang          (9) ');
end;

procedure limpa_funcoes;
begin
     window(3,18,70,24);
     clrscr;
     window(1,1,80,25);
end;

procedure imprime_parametros(F_prob,coluna,linha:byte;var N_parametros:byte);
begin
     if (F_prob)-1) and (F_prob(10) then
     begin
          if F_prob=0 then
          begin
               N_parametros:=1;
               gotoxy(coluna,linha);
               write('A probabilidade e constante          ');
               gotoxy(coluna,linha+1);
               write('k                =                   ');
          end;
          if F_prob=1 then
          begin
               N_parametros:=2;
               gotoxy(coluna,linha);
               write('Distribuicao Uniforme                ');
               gotoxy(coluna,linha+1);
               write('a                =                   ');
               gotoxy(coluna,linha+2);
               write('b                =                   ');
          end;
          if F_prob=2 then
          begin
               N_parametros:=1;
               gotoxy(coluna,linha);
               write('Distribuicao Exponencial             ');
               gotoxy(coluna,linha+1);
               write('media            =                   ');
          end;
          if F_prob=3 then
          begin
               N_parametros:=2;
               gotoxy(coluna,linha);
```

```
            write('Distribuicao Gama                ');
            gotoxy(coluna,linha+1);
            write('alfa             =               ');
            gotoxy(coluna,linha+2);
            write('numero de eventos =              ');
        end;
        if F_prob=4 then
        begin
            N_parametros:=2;
            gotoxy(coluna,linha);
            write('Distribuicao Normal              ');
            gotoxy(coluna,linha+1);
            write('media            =               ');
            gotoxy(coluna,linha+2);
            write('desvio padrao    =               ');
        end;
        if F_prob=5 then
        begin
            N_parametros:=2;
            gotoxy(coluna,linha);
            write('Distribuicao Binomial Negativa   ');              gotoxy(coluna,linha+1);
            write('numero de sucessos=              ');
            gotoxy(coluna,linha+2);
            write('q                =               ');
        end;
        if F_prob=6 then
        begin
            N_parametros:=2;
            gotoxy(coluna,linha);
            write('Distribuicao Binomial            ');
            gotoxy(coluna,linha+1);
            write('numero de ensaios =              ');
            gotoxy(coluna,linha+2);
            write('prob. de sucesso  =              ');
        end;
        if F_prob=7 then
        begin
            N_parametros:=3;
            gotoxy(coluna,linha);
            write('Distribuicao Hipergeometrica     ');
            gotoxy(coluna,linha+1);
            write('tamanho da amostra=              ');
            gotoxy(coluna,linha+2);
            write('tamanho populacao =              ');
            gotoxy(coluna,linha+3);
            write('prob. de sucesso  =              ');
        end;
        if F_prob=8 then
        begin
            N_parametros:=1;
            gotoxy(coluna,linha);
            write('Distribuicao Poisson             ');
            gotoxy(coluna,linha+1);
            write('lambda           =               ');
        end;
        if F_prob=9 then
        begin
            N_parametros:=2;
            gotoxy(coluna,linha);
            write('Distribuicao m-Erlang            ');
            gotoxy(coluna,linha+1);
            write('m                =               ');
            gotoxy(coluna,linha+2);
            write('media            =               ');
        end;
    end;
end;

procedure apaga_parametros;
begin
    window(3,18,70,24);
    clrscr;
    window(1,1,80,25);
end;

procedure imprime_opcoes;
begin
    gotoxy(4,12);
    write('Id opcao de maquina:             ');
    gotoxy(4,13);
```

```
     write('Funcao que descreve o tempo de operacao:              ');
     gotoxy(4,14);
     write('2& opcao de maquina:                  ');
     gotoxy(4,15);
     write('Funcao que descreve o tempo de operacao:              ');
     gotoxy(4,16);
     write('3& opcao de maquina:                  ');
     gotoxy(4,17);
     write('Funcao que descreve o tempo de operacao:              ');
end;

procedure pede_parametros;
Var
   resposta2:string[3];
begin
     resposta2:='n';
     while (resposta2='n') or (resposta2='N') do
     begin
          LeTeclado(25,20,10,'R','4',StrGenerica,RealGenerico,Resultado);
          Parametro1:=0;
          if (Resultado=0) then
            Parametro1:=RealGenerico;
          if N_parametros)1 then
          begin
               LeTeclado(25,21,10,'R','4',StrGenerica,RealGenerico,Resultado);
               Parametro2:=0;
               if (Resultado=0) then
                  Parametro2:=RealGenerico;
          end;
          if N_parametros)2 then
          begin
               LeTeclado(25,22,10,'R','4',StrGenerica,RealGenerico,Resultado);
               Parametro3:=0;
               if (Resultado=0) then
                  Parametro3:=RealGenerico;
          end;
          resposta2:='';
          while((resposta2()'S') and (resposta2()'N')) do
          begin
               gotoxy(56,20);
               write('Tudo OK (S/N);        ');
               LeTeclado(71,20,1,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) then
                   resposta2:=StrGenerica;
          end;
          gotoxy(56,20);
          write('                      ');
     end;
end;
end.(Fim da Unit)
{$O+,F+}
Unit Le_funcoes;
Interface
Uses Dos,Overlay, Crt, Funcoes2, Letecla,Tela, Varler;
{$O Funcoes2}
{$O Tela}
     procedure le_func_prob_quebra;
     procedure le_func_prob_reparo;
     procedure le_F_prob_T_cheg;
     procedure le_F_prob_Tam_lote;
     procedure le_parametros_Tam_lote;
     procedure le_parametros_T_cheg;
Implementation
procedure le_func_prob_quebra;
begin
     resposta:='';
     while(resposta()'S') and (resposta()'s') do
     begin
          F_prob_quebra:=10;
          while(F_prob_quebra(0) or (F_prob_quebra)9) do
          begin
               gotoxy(5,14);
               write('Funcao de probabilidade de quebra da maquina:     ');
               funcoes;
               moldura(3,18,61,24);
               LeTeclado(52,14,2,'B',' ',StrGenerica,RealGenerico,Resultado);
               F_prob_quebra:=0;
               if (Resultado=0) then
               F_prob_quebra:=trunc(RealGenerico);
           end;
```

```
          resposta:='';
          while((resposta()'S') and (resposta()'N')) do
          begin
               gotoxy(5,16);
               write('Tudo OK (S/N);       ');
               LeTeclado(19,16,1,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) then
                  resposta:=StrGenerica;
          end;
          gotoxy(5,16);
          write('                          ');
          limpa_funcoes;
     end;
     apaga_parametros;
     imprime_parametros(F_prob_quebra,6,19,N_parametros);
     moldura(5,18,37,23);
     pede_parametros;
     apaga_parametros;
end;
procedure le_func_prob_reparo;
begin
     resposta:='';
     while (resposta()'S') and (resposta()'s') do
     begin
          F_prob_reparo:=10;
          while(F_prob_reparo(0) or (F_prob_reparo)9) do
          begin
               gotoxy(5,15);
               writeln('Funcao de probabilidade de reparo da maquina:       ');
               funcoes;
               moldura(3,18,61,24);
               LeTeclado(52,15,2,'B',' ',StrGenerica,RealGenerico,Resultado);
               F_prob_reparo:=0;
               if (Resultado=0) then
               F_prob_reparo:=trunc(RealGenerico);
          end;
          resposta:='';
          while((resposta()'S') and (resposta()'N')) do
          begin
               gotoxy(5,16);
               write('Tudo OK (S/N);       ');
               LeTeclado(19,16,1,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) then
                  resposta:=StrGenerica;
          end;
          gotoxy(5,16);
          write('                          ');
          limpa_funcoes;
     end;
     apaga_parametros;
     imprime_parametros(F_prob_reparo,6,19,N_parametros);
     moldura(5,18,37,23);
     pede_parametros;
     apaga_parametros;
     window(3,5,75,23);
     clrscr;
     window(1,1,80,25);
end;

procedure le_F_Prob_T_cheg;
var
   chave:string[3];
begin
     funcoes;
     moldura(3,18,61,24);
     chave:='n';
     while (chave='n') or (chave='N') do
     begin
          LeTeclado(57,5,1,'B',' ',StrGenerica,RealGenerico,Resultado);
          F_prob_tempo_chegada:=0;
          if (Resultado=0) then
             F_prob_tempo_chegada:=trunc(RealGenerico);
          gotoxy(57,5);
          write('      ');
          gotoxy(57,5);
          write(F_prob_tempo_chegada);
          if (F_prob_tempo_chegada)-1) and (F_prob_tempo_chegada(10) then chave:='s';
          end;
          limpa_funcoes;
          resposta:='';
```

```
        while((resposta()'S') and (resposta()'N')) do
        begin
             gotoxy(4,7);
             write('Tudo OK (S/N):      ');
             LeTeclado(19,7,1,'S','U',StrGenerica,RealGenerico,Resultado);
             if (Resultado=0) then
                resposta:=StrGenerica;
        end;
        gotoxy(4,7);
       writeln('                          ');
end;

procedure le_parametros_T_cheg;
begin
     apaga_parametros;
     imprime_parametros(F_prob_tempo_chegada,6,19,N_parametros);
     moldura(5,18,37,23);
     pede_parametros;
     gotoxy(56,20);
     writeln('                ');
     apaga_parametros;
end;

procedure le_F_Prob_Tam_lote;
var
   chave:string[3];
begin
     limpa_funcoes;
     funcoes;
     moldura(3,18,61,24);
     resposta2:='n';
     while (resposta2='n') or (resposta2='N') do
     begin
          chave:='n';
          while (chave='n') or (chave='N') do
          begin
               gotoxy(3,6);
               writeln('F.prob. p/ tamanho do lote:          ');
               LeTeclado(31,6,3,'B',' ',StrGenerica,RealGenerico,Resultado);
               F_prob_tamanho_lote:=0;
               if (Resultado=0) then
                  F_prob_tamanho_lote:=trunc(RealGenerico);
               if (F_prob_tamanho_lote)-i) and (F_prob_tamanho_lote(10) then chave:='s';
          end;
          resposta2:='';
          while((resposta2()'S') and (resposta2()'N')) do
          begin
               gotoxy(3,9);
               writeln('Tudo OK (S/N):       ');
               LeTeclado(18,9,1,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) then
                  resposta2:=StrGenerica;
          end;
     end;
end;

procedure le_parametros_tam_lote;
begin
     limpa_funcoes;
     N_parametros:=0;
     apaga_parametros;
     imprime_parametros(F_prob_tamanho_lote,6,19,N_parametros);
     moldura(5,18,42,23);
     pede_parametros;
     gotoxy(3,9);
     writeln('                      ');
     limpa_funcoes;
     apaga_parametros;
end;
end.
{$O+,F+}
Unit Dist;
Interface
Uses Dos, Overlay,Crt, Defdist, Tela, Le_funcoes, Letecla, Defenti, Defent2, VarLer;
{$O Defdist}
{$O Tela}
{$O Le_funcoes}
{$O Defenti}
{$O Defent2}
     procedure distancias;
```

```
Implementation

procedure distancias;
begin
     clrscr;
     moldura2(2,2,79,25);
     window(3,3,78,24);
     clrscr;
     window(1,1,80,25);
     resposta:=' ';
     while (resposta()'s') and (resposta()'N') and (resposta()'n') and (resposta()'S') do
     begin
          while (resposta()'s') and (resposta()'N') and (resposta()'S') and (resposta()'n') and (resposta()'J')
          do
            begin
                 gotoxy(15,5);
                 write('TECLE (J) PARA AJUDA');
                 gotoxy(4,10);
                 write('Deseja entrar com coordenadas das celulas ((S)im/(N)ao/a(J)uda):     ');
                 LeTeclado(68,10,1,'S','U',StrGenerica,RealGenerico,Resultado);
                 if Resultado=0 then resposta:=StrGenerica;
            end;
            if (resposta='J') then
            begin
            end;
     end;
     window(3,3,78,24);
     clrscr;
     window(1,1,80,25);
     if (resposta='s') or (resposta='S') then
       begin
            resp:='';
            while(resp()'S') do
            begin
                 drive_arquivo:='';
                 while (drive_arquivo='') do
                 begin
                      gotoxy(5,3);
                      write('Drive do arquivo das coordenadas:                                    ');
                      LeTeclado(39,3,30,'S','U',StrGenerica,RealGenerico,Resultado);
                      drive_arquivo:=StrGenerica;
                 end;
                 gotoxy(5,5);
                 write('Nome do arquivo das coordenadas:                                    ');
                 LeTeclado(38,5,12,'S','U',StrGenerica,RealGenerico,Resultado);
                 Nome_arquivo_coord:=' ';
                 if (Resultado=0) then
                 Nome_arquivo_coord:=StrGenerica;
                 Nome_arquivo_coord:=drive_arquivo+'\'+Nome_arquivo_coord;
                 gotoxy(3,3);
                 write('                                                                    ');
                 gotoxy(38,5);
                 write('                              ');
                 gotoxy(38,5);
                 write(Nome_arquivo_coord);
                 resp:='';
                 while(resp()'S') and (resp()'N') do
                 begin
                      gotoxy(5,10);
                      write('Tudo OK (S/N);       ');
                      LeTeclado(23,10,1,'S','U',StrGenerica,RealGenerico,Resultado);
                      if (Resultado=0) then
                      resp:=StrGenerica;
                 end;
            end;
            ASSIGN(Var2deArquivo,Nome_arquivo_coord);
            REWRITE(Var2deArquivo);
            window(3,3,78,24);
            clrscr;
            window(1,1,80,25);
            entr_coord_saida;   (*** ENTRADA DAS COORDENADAS DA SAIDA ***)
            entr_coord_armaz;   (*** ENTRADA DAS COORDENADAS DO ARMAZENAMENTO ***)
            entr_coord_cel;     (*** ENTRADA DAS COORDENADAS DAS CELULAS *** )
             window(3,3,78,24);
             clrscr;
             window(1,1,80,25);
             gotoxy(10,3);
             write('ENTRADA DE COORDENADAS DE MAQUINAS DENTRO DAS CELULAS');
             for ii:=1 to N_celulas do
             begin
```

```
                entr_coord_maq;
            end;
            CLOSE(Var2deArquivo);
        end;
end;
end.
{$O+,F+}
Unit Defent1;
Interface
Uses Dos,Overlay,Crt, Defvar,varler, Tela, le_funcoes, funcoes2, Letecla;
{$O Tela}
{$O le_funcoes}
{$O Funcoes2}
    procedure entrada_sistema_transferencia;
    procedure informacoes_iniciais;
    procedure gravacao_sistema_transferencia;
    procedure le_ident_e_nmaq_de_celula(N_cel:byte);
    procedure le_caract_de_maquina;
Implementation
procedure entrada_sistema_transferencia;
begin
    window(3,3,78,24);
    clrscr;
    window(1,1,80,25);
    resposta2:='n';
    while (resposta2='n') or (resposta2='N') do
    begin
        gotoxy(5,10);
        write('Nome do Sistema de Transferencia:                                    ');
        Nome_sistema_transferencia:='';
        LeTeclado(39,10,30,'S','U',StrGenerica,RealGenerico,Resultado);
        if (Resultado=0) then
           Nome_sistema_transferencia:=StrGenerica;
        gotoxy(39,10);
        write('                                ');
        gotoxy(39,10);
        write(Nome_sistema_transferencia);
        gotoxy(5,15);
        writeln('Velocidade media de atuacao:                       ');
        LeTeclado(34,15,10,'R','4',StrGenerica,RealGenerico,Resultado);
        Velocidade_sistema_transferencia:=0;
        if (Resultado=0) then
           Velocidade_sistema_transferencia:=RealGenerico;
        gotoxy(34,15);
        write('            ');
        gotoxy(34,15);
        write(Velocidade_sistema_transferencia);
        gotoxy(5,18);
        writeln('Numero de equipamentos do Sistema de Transferencia:          ');
        LeTeclado(57,18,3,'B',' ',StrGenerica,RealGenerico,Resultado);
        N_equip_Sist_transf:=0;
        if (Resultado=0) then
           N_equip_Sist_transf:=trunc(RealGenerico);
        resposta2:='';
        while((resposta2()'S') and (resposta2()'N')) do
        begin
            gotoxy(5,22);
            write('Tudo OK (S/N):          ');
            LeTeclado(20,22,1,'S','U',StrGenerica,RealGenerico,Resultado);
            if (Resultado=0) then
               resposta2:=StrGenerica;
    end;
end;
procedure gravacao_sistema_transferencia;
begin
    WRITE(VardeArquivo,Nome_sistema_transferencia);
     WRITELN(VardeArquivo,' (- Nome do sistema de transferencia ');
    WRITE(VardeArquivo,Velocidade_sistema_transferencia);
     WRITELN(VardeArquivo,' (- Velocidade media do sistema de transferencia');
    WRITE(VardeArquivo,N_equip_Sist_transf);
    WRITELN(VardeArquivo,' (- Numero de equipamentos do sistema de transferencia ');
end;
procedure informacoes_iniciais;
begin
    clrscr;
    moldura2(2,2,79,25);
    gotoxy(31,1);
    write('CRIACAO DE ARQUIVOS');
```

```
     resposta:='n';
     while (resposta='n') or (resposta='N') do
     begin
          primeira_tela;
          drive_arquivo:='';
          Nome_arquivo:='';
          while(drive_arquivo='') do
          begin
               LeTeclado(29,3,30,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) THEN
               drive_arquivo:=StrGenerica;
          end;
          while(Nome_arquivo='') do
          begin
               LeTeclado(28,5,12,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) THEN
               Nome_arquivo:=StrGenerica;
          end;
          Nome_arquivo:=drive_arquivo+'\'+Nome_arquivo;
          gotoxy(3,3);
          write('                                                    ');
          gotoxy(28,5);
          write('          ');
          gotoxy(28,5);
          write(Nome_arquivo);
          LeTeclado(29,10,30,'S','U',StrGenerica,RealGenerico,Resultado);
          Nome_da_simulacao:='';
          if (Resultado=0) THEN
             Nome_da_simulacao:=StrGenerica;
          gotoxy(29,10);
          write('                                    ');
          gotoxy(29,10);
          write(Nome_da_Simulacao);
          LeTeclado(29,15,3,'B',' ',StrGenerica,RealGenerico,Resultado);
          N_celulas:=0;
          if (Resultado=0) THEN
             N_celulas:=Trunc(RealGenerico);
          gotoxy(29,15);
          write('   ');
          gotoxy(29,15);
          write(N_celulas);
          resposta:='';
          while((resposta()'N') and (resposta()'S')) do
          begin
               gotoxy(10,20);
               write('Esta tudo correto (S/N): ');
               LeTeclado(37,20,1,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) THEN
                  resposta:=StrGenerica;
          end;
end;
procedure le_ident_e_nmaq_de_celula(N_cel:byte);
begin
     window(3,3,78,24);
     clrscr;
     window(1,1,80,25);
     resposta:='n';
     while (resposta='n') or (resposta='N') do
     begin
          segunda_tela(i);
          Ident_celula:=N_cel;
          {LeTeclado(51,3,3,'B',' ',StrGenerica,RealGenerico,Resultado);
          Ident_celula:=0;
          if (Resultado=0) then
             Ident_celula:=trunc(RealGenerico);
          gotoxy(51,3);
          write('   ');}
          gotoxy(51,3);
          write(Ident_celula);
          LeTeclado(25,4,3,'B',' ',StrGenerica,RealGenerico,Resultado);
          Numero_maquinas[i]:=0;
          if (Resultado=0) then
          Numero_maquinas[i]:=trunc(RealGenerico);
          gotoxy(25,4);
          write('   ');
          gotoxy(25,4);
          write(Numero_maquinas[i]);
          resposta:='';
          while((resposta()'S') and (resposta()'N')) do
          begin
```

```
                gotoxy(5,6);
                write('Tudo OK (S/N):     ');
                LeTeclado(20,6,1,'S','U',StrGenerica,RealGenerico,Resultado);
                if (Resultado=0) then
                resposta:=StrGenerica;
            end;
        end;
        gotoxy(5,6);
        writeln('                                        ');

end;
procedure le_caract_de_maquina;
begin
     resposta:='n';
     while (resposta='n') or (resposta='N') do
     begin
          gotoxy(5,6);
          writeln('Maquina ',ii);
          seg_tela_2;
          LeTeclado(22,7,10,'R','2',StrGenerica,RealGenerico,Resultado);
          Tempo_setup:=0;
          if (Resultado=0) then
               Tempo_setup:=RealGenerico;
          Nome_maquina:='  ';
          LeTeclado(23,8,30,'S','A',StrGenerica,RealGenerico,Resultado);
          if (Resultado=0) then
              Nome_maquina:=StrGenerica;
          gotoxy(23,8);
          write('                                  ');
          gotoxy(23,8);
          write(Nome_maquina);
          LeTeclado(49,9,3,'B',' ',StrGenerica,RealGenerico,Resultado);
          Ident_maquina:=0;
          if (Resultado=0) then
              Ident_maquina:=trunc(RealGenerico);
          LeTeclado(45,10,2,'B',' ',StrGenerica,RealGenerico,Resultado);
          Cond_saturacao:=0;
          if (Resultado=0) then
             Cond_saturacao:=trunc(RealGenerico);
          resposta:='  ';
          while((resposta()'S') and (resposta()'N')) do
          begin
               gotoxy(5,12);
               write('Tudo OK (S/N):        ');
               LeTeclado(20,12,1,'S','U',StrGenerica,RealGenerico,Resultado);
               resposta:='  ';
               if (Resultado=0) then
                  resposta:=StrGenerica;
          end;
          gotoxy(5,12);
          write('                          ');
     end;
end;
end.
{$O+,F+}
Unit Defent2;
Interface
Uses Dos,Overlay,Crt, Defvar,varler, Tela, le_funcoes, funcoes2, Letecla;
{$O Tela}
{$O le_funcoes}
{$O Funcoes2}
     procedure le_manipulador;
     procedure le_numero_componentes;
     procedure le_nome_componente;
     procedure le_prioridade_componente;
     procedure le_ident_componente;
     procedure le_numero_operacoes;
     procedure le_operacao;

Implementation
procedure le_manipulador;
begin
     resposta:='n';
     while (resposta='n') or (resposta='N') do
     begin
          seg_tela_3;
          LeTeclado(26,10,30,'S','A',StrGenerica,RealGenerico,Resultado);
          Nome_manipulador:='  ';
          if (Resultado=0) then
             Nome_manipulador:=StrGenerica;
```

```
            gotoxy(26,10);
            write('                                ');
            gotoxy(26,10);
            write(Nome_manipulador);
            LeTeclado(49,14,10,'R','2',StrGenerica,RealGenerico,Resultado);
            Velocidade_manipulador:=0;
            if (Resultado=0) then
               Velocidade_manipulador:=RealGenerico;
            resposta:=' ';
            while((resposta()'S') and (resposta()'N')) do
            begin
                 gotoxy(5,18);
                 write('Tudo OK (S/N):        ');
                 LeTeclado(20,18,1,'S','U',StrGenerica,RealGenerico,Resultado);
                 if (Resultado=0) then
                    resposta:=StrGenerica;
            end;
       end;
  end;
  procedure le_numero_componentes;
  begin
       window(3,3,78,24);
       clrscr;
       window(1,1,80,25);
       resposta:='n';
       while (resposta='n') or (resposta='N') do
       begin
            terceira_tela;
            LeTeclado(33,10,3,'B',' ',StrGenerica,RealGenerico,Resultado);
            N_componentes:=0;
            if (Resultado=0) then
               N_componentes:=trunc(RealGenerico);
            gotoxy(33,10);
            write('      ');
            gotoxy(33,10);
            write(N_componentes);
            resposta:=' ';
            while((resposta()'S') and (resposta()'N')) do
            begin
                 gotoxy(10,12);
                 write('Tudo OK (S/N):      ');
                 LeTeclado(25,12,1,'S','U',StrGenerica,RealGenerico,Resultado);
                 if (Resultado=0) then
                    resposta:=StrGenerica;
            end;
       end;
       window(3,3,78,24);
       clrscr;
       window(1,1,80,25);
  end;
  procedure le_nome_componente;
  begin
       gotoxy(4,3);
       writeln('Nome do componente ',i2,':                          ');
       gotoxy(4,4);
       writeln('Identificacao do componente ',i2,':              ');
       gotoxy(4,5);
       writeln('F. prob. p/ tempo de chegada de elemento no sistema:       ');
       LeTeclado(26,3,30,'S','A',StrGenerica,RealGenerico,Resultado);
       Nome_componente:=' ';
       if (Resultado=0) then
          Nome_componente:=StrGenerica;
       gotoxy(26,3);
       write('                              ');
       gotoxy(26,3);
       write(Nome_componente);
       limpa_funcoes;
  end;
  procedure le_prioridade_componente;
  begin
       gotoxy(3,7);
       writeln('Prioridade do componente:           ');
       prioridade_componente:=0;
       LeTeclado(29,7,3,'B',' ',StrGenerica,RealGenerico,Resultado);
       if (Resultado=0) then
         prioridade_componente:=trunc(RealGenerico);
  end;
  procedure le_ident_componente;
  begin
      LeTeclado(42,4,3,'B',' ',StrGenerica,RealGenerico,Resultado);
```

```
     Ident_componente:=0;
     if (Resultado=0) then
        Ident_componente:=trunc(RealGenerico);
end;
procedure le_numero_operacoes;
begin
     gotoxy(3,9);
     writeln('Numero de operacoes a executar em maquina:          ');
     LeTeclado(46,9,3,'B',   ,StrGenerica,RealGenerico,Resultado);
     N_operacoes:=0;
     if (Resultado=0) then
        N_operacoes:=trunc(RealGenerico);
     gotoxy(46,9);
     write(       );
     gotoxy(46,9);
     write(N_operacoes);
end;
procedure le_operacao;
var
   chave:string[3];
begin
     gotoxy(3,10);
     writeln('Operacao ',i3);
     imprime_opcoes;
     resposta2:='n';
     while (resposta2='n') or (resposta2='N') do
     begin
          LeTeclado(24,12,3,'B',' ',StrGenerica,RealGenerico,Resultado);
          opcao_1_maquina:=0;
          if (Resultado=0) then
              opcao_1_maquina:=trunc(RealGenerico);
          gotoxy(24,12);
          write(       );
          gotoxy(24,12);
          write(opcao_1_maquina);
          limpa_funcoes;
          funcoes;
          moldura(3,18,61,24);
          chave:='n';
          while (chave='n') or (chave='N') do
          begin
               LeTeclado(44,13,3,'B',' ',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) then
                  F_prob_opcao_1:=trunc(RealGenerico);
               gotoxy(44,13);
               write(       );
               gotoxy(44,13);
               write(F_prob_opcao_1);
               if (F_prob_opcao_1)-1) and (F_prob_opcao_1(10) then chave:='s';
          end;
          limpa_funcoes;
          apaga_parametros;
          imprime_parametros(F_prob_opcao_1,6,19,N_parametros);
          moldura(5,18,42,23);
          chave:='n';
          while (chave='n') or (chave='N') do
          begin
              LeTeclado(25,20,10,'R','4',StrGenerica,RealGenerico,Resultado);
              if (Resultado=0) then
                 par11:=RealGenerico;
              if N_parametros)1 then
              begin
                   LeTeclado(25,21,10,'R','4',StrGenerica,RealGenerico,Resultado);
                   if (Resultado=0) then
                      par12:=RealGenerico;
                   end;
                   if N_parametros)2 then
                   begin
                        LeTeclado(25,22,10,'R','4',StrGenerica,RealGenerico,Resultado);
                        if (Resultado=0) then
                           par13:=RealGenerico;
                   end;
                   chave:='';

                          while((chave()'S') and (chave()'N')) do
                          begin
                               gotoxy(56,20);
                               write('Tudo OK (S/N):      ');
                               LeTeclado(71,20,1,'S','U',StrGenerica,RealGenerico,Resultado);
                               if (Resultado=0) then
```

```
            chave:=StrGenerica;
        end;
    end;
    gotoxy(56,20);
    write('                    ');
    N_par1:=N_parametros;
    limpa_funcoes;
    apaga_parametros;
    LeTeclado(24,14,3,'B',' ',StrGenerica,RealGenerico,Resultado);
    if (Resultado=0) then
       opcao_2_maquina:=trunc(RealGenerico);
    gotoxy(24,14);
    write('      ');
    gotoxy(24,14);
    write(opcao_2_maquina);
    limpa_funcoes;
    funcoes;
    moldura(3,18,61,24);
    chave:='n';
    while (chave='n') or (chave='N') do
    begin
        LeTeclado(44,15,3,'B',' ',StrGenerica,RealGenerico,Resultado);
        if (Resultado=0) then
           F_prob_opcao_2:=trunc(RealGenerico);
        gotoxy(44,15);
        write('      ');
        gotoxy(44,15);
        write(F_prob_opcao_2);
        if (F_prob_opcao_2)-1) and (F_prob_opcao_2(10) then chave:='s';
    end;
    limpa_funcoes;
    apaga_parametros;
    imprime_parametros(F_prob_opcao_2,6,19,N_parametros);
    moldura(5,18,42,23);
    chave:='n';
    while (chave='n') or (chave='N') do
    begin
        LeTeclado(25,20,10,'R','4',StrGenerica,RealGenerico,Resultado);
        if (Resultado=0) then
           par21:=RealGenerico;
        if N_parametros)1 then
        begin
            LeTeclado(25,21,10,'R','4',StrGenerica,RealGenerico,Resultado);
            if (Resultado=0) then
               par22:=RealGenerico;
        end;
        if N_parametros)2 then
        begin
            LeTeclado(25,22,10,'R','4',StrGenerica,RealGenerico,Resultado);
            if (Resultado=0) then
               par23:=RealGenerico;
        end;
        chave:='';
        while((chave()'S') and (chave()'N')) do
        begin
            gotoxy(56,20);
            write('Tudo OK (S/N);      ');
            LeTeclado(71,20,1,'S','U',StrGenerica,RealGenerico,Resultado);
            if (Resultado=0) then
            chave:=StrGenerica;
        end;
    end;
    gotoxy(56,20);
    write('                    ');
    N_par2:=N_parametros;
    limpa_funcoes;
    apaga_parametros;
    LeTeclado(24,16,3,'B',' ',StrGenerica,RealGenerico,Resultado);
    if (Resultado=0) then
    opcao_3_maquina:=trunc(RealGenerico);
    gotoxy(24,16);
    write('      ');
    gotoxy(24,16);
    write(opcao_3_maquina);
    limpa_funcoes;
    funcoes;
    moldura(3,18,61,24);
    chave:='n';
    while (chave='n') or (chave='N') do
    begin
```

```
                    LeTeclado(44,17,3,'B',' ',StrGenerica,RealGenerico,Resultado);
                    if (Resultado=0) then
                       F_prob_opcao_3:=trunc(RealGenerico);
                    gotoxy(44,17);
                    write('    ');
                    gotoxy(44,17);
                    write(F_prob_opcao_3);
                    if (F_prob_opcao_3)-1) and (F_prob_opcao_3(10) then chave:='s';
                end;
                apaga_parametros;
                imprime_parametros(F_prob_opcao_3,6,19,N_parametros);
                moldura(5,18,42,23);
                chave:='n';
                while (chave='n') or (chave='N') do
                begin
                    LeTeclado(25,20,10,'R','4',StrGenerica,RealGenerico,Resultado);
                    if (Resultado=0) then
                       par31:=RealGenerico;
                    if N_parametros)1 then
                    begin
                        LeTeclado(25,21,10,'R','4',StrGenerica,RealGenerico,Resultado);
                        if (Resultado=0) then
                           par32:=RealGenerico;
                    end;
                    if N_parametros)2 then
                    begin
                        LeTeclado(25,22,10,'R','4',StrGenerica,RealGenerico,Resultado);
                        if (Resultado=0) then
                           par33:=RealGenerico;
                    end;
                    chave:='';
                    while((chave()'S') and (chave()'N')) do
                    begin
                        gotoxy(56,20);
                        write('Tudo OK (S/N):     ');
                        LeTeclado(71,20,1,'S','U',StrGenerica,RealGenerico,Resultado);
                        if (Resultado=0) then
                   chave:=StrGenerica;
                end;
            end;
            gotoxy(56,20);
            write('                    ');
            N_par3:=N_parametros;
            limpa_funcoes;
            apaga_parametros;
            resposta2:='';
            while((resposta2()'S') and (resposta2()'N')) do
            begin
                gotoxy(3,19);
                write('Tudo OK (S/N):       ');
                LeTeclado(18,19,1,'S','U',StrGenerica,RealGenerico,Resultado);
                if (Resultado=0) then
                   resposta2:=StrGenerica;
            end;
    end;
end;
end.
($0+,F+)
Unit learq;
Interface
uses dos,overlay,crt,varler,ler,coord,defvar,lista_celula,
     lista_job,lista_tarefa,letecla,tela,prob;
        procedure leitura_arquivo;
Implementation
procedure le(var cadeia2:sentenca);
var
  CH:string[1];
  chv:boolean;
  cadeia:sentenca;
begin
    cadeia:='';
    CH:='';
    while (chv=false) do
    begin
        Cadeia:=Cadeia+CH;
        READ(ArqEntrada,CH);
        if CH='*' then chv:=true;
    end;
    cadeia2:=cadeia;
    READLN(ArqEntrada);
```

```
end;
procedure leitura_arquivo;
var
   le_celula:ptr_celula;
   le_job:ptr_job;
   le_componente:ponteiro;
   tempo:real;
   Tamanho:word;
   Cadeia:sentenca;
begin
     clrscr;
     moldura(2,2,79,25);
     gotoxy(31,1);
     write('LEITURA DE ARQUIVOS');
     resposta:='n';
     while (resposta='n') or (resposta='N') do
     begin
          primeira_tela_leitura;
          drive_arquivo:='';
          while(drive_arquivo='') do
          begin
               LeTeclado(29,3,30,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) THEN
               drive_arquivo:=StrGenerica;
          end;
          Nome_arquivo:='';
          while(Nome_arquivo='') do
          begin
               LeTeclado(28,5,12,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) THEN
               Nome_arquivo:=StrGenerica;
          end;
          Nome_arquivo:=drive_arquivo+'\'+Nome_arquivo;
          gotoxy(3,3);
          write('                                                    ');
          gotoxy(28,5);
          write('          ');
          gotoxy(28,5);
          write(Nome_arquivo);
          resposta:='';
          while((resposta()'N') and (resposta()'S')) do
          begin
               gotoxy(10,20);
               write('Esta tudo correto:        ');
               LeTeclado(29,20,1,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) then
                  resposta:=Strgenerica;
          end;
     end;
     ASSIGN(ArqEntrada,Nome_arquivo);
     RESET(ArqEntrada);
     LE(Cadeia);
     Nome_da_simulacao:=Cadeia;
     READLN(ArqEntrada,Num_celulas);
     inicia_lista;
     for i:=1 to Num_celulas do
     begin
          new (le_celula);
          READLN(ArqEntrada,le_celula^.Ident_celula);
          READLN(ArqEntrada,le_celula^.N_maquinas);
          Numero_maquinas[i]:=le_celula^.N_maquinas;
          with le_celula^ do
          begin
               for ii:=1 to N_maquinas DO
               begin
                    with maquina[ii] do
                    begin
                         T_set_up:=0;
                         Name:='';
                         Ident_maquina:=0;
                         READLN(ArqEntrada,T_set_up);
                    end;
                    le(Cadeia);
                    maquina[ii].Name:=Cadeia;
                    with maquina[ii] do
                    begin
                         READLN(ArqEntrada,Ident_maquina);
                         READLN(ArqEntrada,Cond_saturacao);
                         READLN(ArqEntrada,Distr_quebra);
                         acha_N_parametros(Distr_quebra,N_par1);
```

```
                    READLN(ArqEntrada,par_quebra1);
                    if N_par1>1 then
                       READLN(ArqEntrada,par_quebra2);
                    if N_par1>2 then
                       READLN(ArqEntrada,par_quebra3);
                    READLN(ArqEntrada,Dist_reparo);
                   acha_N_parametros(Dist_reparo,N_par1);
                    READLN(ArqEntrada,par_rep1);
                    if N_par1>1 then
                       READLN(ArqEntrada,par_rep2);
                    if N_par1>2 then
                       READLN(ArqEntrada,par_rep3);
               end;
          end;
          with manip do
          begin
               Nome:='';
               Veloc:=0;
               Nome:=Cadeia;
               READLN(ArqEntrada,Veloc);
          end;
     end;
     insere_celula(raiz_celula,le_celula);
end;
Num_componentes:=0;
READLN(ArqEntrada,Num_componentes);
for i2:=1 to Num_componentes DO
begin
     getmem(le_componente,sizeof(tarefa));
     new(le_job);
     begin
          le_job^.componente:='';
          le_componente^.rota:=1;
          le_componente^.saida:=false;
          le_componente^.operador:=0;
          le_componente^.T_ocupacao:=0.0;
          le_componente^.pos_atual:=0;
          le_componente^.maq:=0;
          le(Cadeia);
          le_job^.componente:=Cadeia;
          READLN(ArqEntrada,Identificacao);
          le_job^.identificacao:=identificacao;
          le_componente^.identificacao:=identificacao;
          READLN(ArqEntrada,le_job^.distr_cheg_armaz);
          acha_N_parametros(le_job^.distr_cheg_armaz,N_parametros);
          READLN(ArqEntrada,le_job^.Par_cheg1);
          if N_parametros>1 then
            READLN(ArqEntrada,le_job^.Par_cheg2);
          if N_parametros>2 then
            READLN(ArqEntrada,le_job^.Par_cheg3);
          tempo_de_chegada(le_job,Tempo);
          le_componente^.T_cheg_armazenamento:=Tempo;
          le_componente^.T_ordenacao:=le_componente^.T_cheg_armazenamento;
     READLN(ArqEntrada,le_job^.distr_lote);
     acha_N_parametros(le_job^.distr_lote,N_parametros);
     READLN(ArqEntrada,le_job^.Par_l1);
     if N_parametros>1 then
        READLN(ArqEntrada,le_job^.Par_l2);
     if N_parametros>2 then
        READLN(ArqEntrada,le_job^.Par_l3);
     tamanho_de_lote(le_job,Tamanho);
     le_componente^.Tam_lote:=Tamanho;
     READLN(ArqEntrada,le_componente^.prioridade);
     Num_tarefas:=0;
     READLN(ArqEntrada,Num_tarefas);
     le_job^.last:=Num_tarefas+1;
     le_componente^.N_op_completar:=Num_tarefas;
     for i3:=1 to Num_tarefas do
     begin
         with le_job^.elemento[i3] do
         begin
              READLN(ArqEntrada,maquina[1]);
              READLN(ArqEntrada,distribuicao[1]);
              READLN(ArqEntrada,maquina[2]);
              READLN(ArqEntrada,distribuicao[2]);
              READLN(ArqEntrada,maquina[3]);
              READLN(ArqEntrada,distribuicao[3]);
              READLN(ArqEntrada,N_par1);
              READLN(ArqEntrada,par_1[1]);
                   if N_par1>1 then
```

```
                    READLN(ArqEntrada,par_2[i]);
                 if N_par1>2 then
                    READLN(ArqEntrada,par_3[i]);
                 READLN(ArqEntrada,N_ppar2);
                 READLN(ArqEntrada,par_1[2]);
                 if N_ppar2>1 then
                    READLN(ArqEntrada,par_2[2]);
                 if N_ppar2>2 then
                    READLN(ArqEntrada,par_3[2]);
                 READLN(ArqEntrada,N_par3);
                 READLN(ArqEntrada,par_1[3]);
                 if N_par3>1 then
                    READLN(ArqEntrada,par_2[3]);
                 if N_par3>2 then
                    READLN(ArqEntrada,par_3[3]);
              end;
        end;
     end;
     cria_lista_ordenada(armazenamento,le_componente);
     cria_lista_jobs(raiz_job,le_job);
end;
le(Cadeia);
Const1:=Cadeia;
READLN(ArqEntrada,Const2);
READLN(ArqEntrada,N_equip_ST);
for i:=1 to N_equip_ST do
begin
     ST[i].tipo:=Const1;
     ST[i].velocidade:=Const2;
end;
gotoxy(10,12);
writeln('LEITURA EFETUADA');
gotoxy(10,15);
writeln('********** FECHANDO ARQUIVO **********');
CLOSE(ArqEntrada);
window(3,3,78,24);
clrscr;
window(1,1,80,25);
resposta:=' ';
while resposta()'S' do
begin
     gotoxy(10,10);
     writeln('Deseja ler arquivo de coordenadas (S/N):       ');
     LeTeclado(51,10,1,'S','U',StrGenerica,RealGenerico,Resultado);
     if Resultado=0 then
        resp:=StrGenerica;
     resposta:=' ';
     while (resposta()'S') and (resposta()'N') do
     begin
          gotoxy(10,15);
          write('Tudo OK (S/N):       ');
          LeTeclado(24,15,1,'S','U',StrGenerica,RealGenerico,Resultado);
          if (Resultado=0) then
             resposta:=StrGenerica;
     end;
end;
if resp='S' then
begin
window(3,3,78,24);
clrscr;
window(1,1,80,25);
resp:=' ';
while (resp()'S') do
begin
     gotoxy(5,3);
     write('Drive do arquivo das coordenadas:                                ');
     LeTeclado(39,3,30,'S','U',StrGenerica,RealGenerico,Resultado);
     Drive_arquivo:=' ';
     if (Resultado=0) THEN
        drive_arquivo:=StrGenerica;
     gotoxy(5,5);
     write('Nome do arquivo das coordenadas:                 ');
     LeTeclado(38,5,12,'S','U',StrGenerica,RealGenerico,Resultado);
     Nome_arquivo_coord:=' ';
     if (Resultado=0) then
        Nome_arquivo_coord:=StrGenerica;
     resp:=' ';
     while (resp()'S') and (resp()'N') do
     begin
          gotoxy(5,10);
```

```
                write('Tudo OK (S/N):       ');
                LeTeclado(23,10,1,'S','U',StrGenerica,RealGenerico,Resultado);
                if (Resultado=0) then
                   resp:=StrGenerica;
            end;
            Nome_arquivo_coord:=drive_arquivo+'\'+Nome_arquivo_coord;
        end;
        ASSIGN(Var2deArquivo,Nome_arquivo_coord);
        RESET(Var2deArquivo);
        le_coord_saida; (** LEITURA DAS COORDENADAS DA SAIDA **)
        le_coord_armaz; (** LEITURA DAS COORDENADAS DO ARMAZENAMENTO **)
        le_coord_cel;   (** LEITURA DAS COORDENADAS DAS CELULAS **)
        for ii:=1 to Num_celulas do
        begin
            le_coord_maq(ii);
        end;
        CLOSE(Var2deArquivo);
        monta_dist_saida;
        monta_dist_celulas;
        monta_dist_maq;
end;
end;
end.(Fim da Unit)
{$O+,F+}
Unit Ler;
Interface
Uses  Dos, Crt, Defvar, Varler;
      procedure acha_N_parametros(F_prob:byte;var N_parametros:byte);
      procedure inicia_lista;
Implementation
procedure acha_N_parametros(F_prob:byte;var N_parametros:byte);
begin
     if (F_prob)-1) and (F_prob(10) then
     begin
          case F_prob of
          0:N_parametros:=1;
          1:N_parametros:=2;
          2:N_parametros:=1;
          3:N_parametros:=2;
          4:N_parametros:=2;
          5:N_parametros:=2;
          6:N_parametros:=2;
          7:N_parametros:=3;
          8:N_parametros:=1;
          9:N_parametros:=2;
          end;
     end;
end;
procedure inicia_lista;
var
   i:byte;
begin
     new (armazenamento);
     armazenamento^.ptr_proximo:=nil;
     new (raiz_celula);
     raiz_celula^.ptr_proximo:=nil;
     for i:=1 to Num_celulas do
     begin
          new (Pont_celula[i]);
          Pont_celula[i]^.ptr_proximo:=nil;
     end;
     new (Pont_ST);
     Pont_ST^.ptr_proximo:=nil;
     new (Raiz_job);
     Raiz_job^.ptr_proximo:=nil;
end;
end.
{$O+,F+}
Unit Coord;
Interface
Uses Crt, Overlay, Ler, defvar, varler;
{$O Ler}

     procedure le_coord_saida;
     procedure le_coord_armaz;
     procedure le_coord_cel;
     procedure le_coord_maq(Cont_celula:byte);
     procedure monta_dist_saida;
     procedure monta_dist_celulas;
     procedure monta_dist_maq;
```

```
Implementation
procedure le_coord_saida;
begin
     READ(Var2deArquivo,dist_saida_x);
     READLN(Var2deArquivo,dist_saida_y);
end;
procedure le_coord_armaz;
begin
     READ(Var2deArquivo,dist_celula[0,1]);
     READLN(Var2deArquivo,dist_celula[0,2]);
end;
procedure le_coord_cel;
var
   cont_celula:byte;
begin
     for cont_celula:=1 to Num_celulas do
     begin
          READ(Var2deArquivo,dist_celula[cont_celula,1]);
          READLN(Var2deArquivo,dist_celula[cont_celula,2]);
     end;
end;
procedure le_coord_maq(Cont_celula:byte);
begin
     for i:=1 to Numero_maquinas[Cont_celula] do
     begin
          READ(Var2deArquivo,coord_maq_incelula[Cont_celula,i,1]);
          READLN(Var2deArquivo,coord_maq_incelula[Cont_Celula,i,2]);
     end;
     coord_maq_incelula[cont_celula,0,2]:=0.0;
     coord_maq_incelula[cont_celula,0,1]:=0.0;
end;
procedure monta_dist_celulas;
var
   cont_celula:byte;
begin
     contador:=0;
     for cont_celula:=0 to Num_celulas do
     begin
          for i2:=contador to Num_celulas do
          begin
               var2:=(dist_celula[cont_celula,1]-dist_celula[i2,1]);
               var3:=(dist_celula[cont_celula,2]-dist_celula[i2,2]);
               var1:=(var2*var2)+(var3*var3);
               Dist_Celulas[cont_celula,i2]:=sqrt(var1);
               Dist_Celulas[i2,cont_celula]:=Dist_Celulas[cont_celula,i2];
          end;
          contador:=contador+1;
     end;
end;
procedure monta_dist_saida;
var
   Cont:byte;
begin
     dist_saida[0]:=sqrt(sqr(dist_celula[0,1]-dist_saida_x)+sqr(dist_celula[0,2]-dist_saida_y));
     for Cont:=1 to Num_celulas do
     begin
          dist_saida[Cont]:=sqrt(sqr(dist_celula[Cont,1]-dist_saida_x)+sqr(dist_celula[Cont,2]-dist_saida_y));
     end;
end;
procedure monta_dist_maq;
var cont_celula:byte;
begin
     for i:=1 to Num_celulas do
     begin
          contador:=0;
          for cont_celula:=0 to Numero_maquinas[i] do
          begin
               for i2:=contador to Numero_maquinas[i] do
               begin
                    var2:=(coord_maq_incelula[i,cont_celula,1]-coord_maq_incelula[i,i2,1]);
                    var3:=(coord_maq_incelula[i,cont_celula,2]-coord_maq_incelula[i,i2,2]);
                    var1:=(var2*var2)+(var3*var3);
                    dist_incelula[i,cont_celula,i2]:=sqrt(var1);
                    dist_incelula[i,i2,cont_celula]:=dist_incelula[i,cont_celula,i2];
               end;
               contador:=contador+1;
          end;
     end;
end;
end.
```

```
Unit Est;
Interface
        Uses Dos, Defvar, Distr, Prob;
        procedure Incrementa_fila_manipulador(N_Cel:byte);
        procedure Decrementa_fila_manipulador(N_Cel:byte);
        procedure Incrementa_fila_maquina(Maquina:byte);
        procedure Decrementa_fila_maquina(Maquina:byte);
        procedure Incrementa_fila_ST;
        procedure Decrementa_fila_ST;
        procedure Incrementa_Espera(TN:ponteiro);
        procedure Est_final_manip(N_celula:byte);
        procedure Est_final_maq(N_celula,Ind_maquina:byte);
        procedure Est_final_ST;
        procedure Est_final_comp(Ident:byte);
        procedure Est_final_grupos(Ident:byte);
Implementation
procedure Incrementa_fila_manipulador;
var
   Subst:real;
begin
     Subst:=T_ult_alt_fila_manip[N_cel];
     Area_sob_curva_manip[N_Cel]:=Area_sob_curva_manip[N_Cel]+N_fila_manipulador[N_Cel]*(relogio-Subst);
     N_fila_Manipulador[N_Cel]:=N_fila_Manipulador[N_Cel]+1;
     T_ult_alt_fila_manip[N_Cel]:=relogio;
end;
procedure Decrementa_fila_manipulador;
var
   Subst:real;
begin
     Subst:=T_ult_alt_fila_manip[N_cel];
     Area_sob_curva_manip[N_Cel]:=Area_sob_curva_manip[N_Cel]+N_fila_manipulador[N_Cel]*(relogio-Subst);
     N_fila_Manipulador[N_Cel]:=N_fila_Manipulador[N_Cel]-1;
     T_ult_alt_fila_manip[N_Cel]:=relogio;
end;
procedure Incrementa_fila_maquina;
var
   Subst:real;
begin
     Subst:=T_ult_alt_fila_maq[maquina];
     Area_sob_curva_maq[Maquina]:=Area_sob_curva_maq[Maquina]+N_fila_maq[Maquina]*(relogio-Subst);
     N_fila_Maq[Maquina]:=N_fila_Maq[Maquina]+1;
     T_ult_alt_fila_maq[Maquina]:=relogio;
end;
procedure Decrementa_fila_maquina;
var
   Subst:real;
begin
     Subst:=T_ult_alt_fila_maq[maquina];
     Area_sob_curva_maq[Maquina]:=Area_sob_curva_maq[Maquina]+N_fila_maq[Maquina]*(relogio-Subst);
     N_fila_Maq[Maquina]:=N_fila_Maq[Maquina]-1;
     T_ult_alt_fila_maq[Maquina]:=relogio;
end;
procedure Incrementa_fila_ST;
var
   Subst:real;
begin
     Subst:=T_ult_alt_fila_ST;
     Area_sob_curva_ST:=Area_sob_curva_ST+N_fila_ST*(relogio-Subst);
     N_fila_ST:=N_fila_ST+1;
     T_ult_alt_fila_ST:=relogio;
end;
procedure Decrementa_fila_ST;
var
   Subst:real;
begin
     Subst:=T_ult_alt_fila_ST;
     Area_sob_curva_ST:=Area_sob_curva_ST+N_fila_ST*(relogio-Subst);
     N_fila_ST:=N_fila_ST-1;
     T_ult_alt_fila_ST:=relogio;
end;
procedure Incrementa_Espera;
var
   Ident_job:byte;
begin
     Ident_job:=TN^.identificacao;
     T_demora_fila[Ident_job]:=T_demora_fila[Ident_job]+(relogio-TN^.T_entrada_fila);
end;
procedure Est_final_manip;
begin
    Perc_ocupado_manip:=T_ocupado_manip[N_celula]/T_simulacao;
```

```
     Perc_util_manip:=T_util_manip[N_celula]/T_simulacao;
     Perc_buffer_manip:=T_buffer_manip[N_celula]/T_simulacao;
     Num_medio_fila_manip:=Area_sob_curva_manip[N_celula]/T_simulacao;
end;
procedure Est_final_maq;
begin
     Perc_ocupacao_maq:=T_ocup_maq[N_celula,Ind_maquina]/T_simulacao;
     Perc_reparo_maq:=T_reparo_maq[N_celula,Ind_maquina]/T_simulacao;
end;
procedure Est_final_ST;
begin
     Perc_ocupado_ST:=T_ocupado_ST/T_simulacao;
     Perc_util_ST:=T_util_ST/T_simulacao;
     Perc_armaz_ST:=T_armaz_ST/T_simulacao;
     Num_medio_fila_ST:=Area_sob_curva_ST/T_simulacao;
end;
procedure Est_final_comp;
begin
     if N_vezes_fila[Ident] = 0
     then T_medio_espera:=0
     else T_medio_espera:=T_demora_fila[Ident]/N_vezes_fila[Ident];
end;
procedure Est_final_grupos;
begin
     Num_medio_fila_maq:=Area_sob_curva_maq[Ident]/T_simulacao;
end;
end.
UNIT distr;
Interface
     procedure dist_uniforme(a,b:real;var x:real);
     procedure dist_exponencial(media:real;var x:real);
     procedure dist_gama(alfa:real;numero_de_eventos:real; var x :real);
     procedure dist_normal(media,desvio_padrao:real; var x:real);
     procedure dist_b_neg(numero_de_sucessos,q:real; var x:real);
     procedure dist_binomial(numero_ensaios,prob_sucesso:real; var x:real);
     procedure dist_hiperg (tamanho_amostra,tamanho_populacao,prob_sucesso:real; var x:real);
     procedure dist_poisson(lambda:real; var x:real);
     procedure dist_m_Erlang( m:real; media:real; var x:real);
Implementation
procedure dist_uniforme;
var
   r:real;
begin
          r:=random;
          x:=a+(b-a)*r;
end;
procedure dist_exponencial;
var
   r:real;
begin
     r:=random;
     x:=-media*ln(r);
end;
procedure dist_gama;
var
   r:real;
   tr:real;
   i, K:integer;
begin
     tr:=1.0;
     for i:=1 to trunc(numero_de_eventos)
       do
       begin
            r:=random;
            tr:=tr*r;
       end;
     x:=-ln(tr)/alfa;
end;
procedure dist_normal;
var
   r:real;
   i:integer;
   soma:real;
begin
     soma:=0.0;
     for i:=1 to 12
         do
         begin
              r:=random;
              soma:=soma+r;
```

```
          end;
       x:=desvio_padrao*(soma-6.0)+media
  end;
  procedure dist_b_neg;
  var
     tr,qr,r,nx:real;
     i:integer;
  begin
       tr:=1.0;
       qr:=ln(q);
       for i:=1 to trunc(numero_de_sucessos)
       do
         begin
              r:=random;
              tr:=tr*r;
              nx:=ln(tr)/qr;
              x:=nx;
         end;
  end;
  procedure dist_binomial;
  var
     i:integer;
     r:real;
  begin
       x:=0.0;
       for i:=1 to trunc(numero_ensaios)
       do
         begin
              r:=random;
              if (r-prob_sucesso)<=0 then x:=x+1.0;
         end;
  end;
  procedure dist_hiperg;
  var
     i:integer;
     r,s:real;
  begin
       x:=0.0;
       for i:=1 to trunc(tamanho_populacao)
       do
         begin
              r:=random;
              if (r-prob_sucesso)>0 then s:=0.0
                 else
                     begin
                          s:=1.0;
                          x:=x+1.0;
                     end;
              prob_sucesso:=(tamanho_amostra*prob_sucesso-s)/(tamanho_amostra-1.0);
              tamanho_amostra:=tamanho_amostra-1;
         end;
  end;
  procedure dist_poisson;
  var
     tr,b,r:real;
     condicao:boolean;
  begin
       x:=0.0;
       tr:=1.0;
       b:=exp(-lambda);
       condicao:=false;
       while condicao=false
       do
         begin
            r:=random;
            tr:=tr*r;
            if (tr-b)<0 then
                 condicao:=true
              else
                 x:=x+1.0;
         end;
  end;
  procedure dist_m_Erlang;
  var
    I:byte;
    U,r:real;
  begin
       U:=1.0;
       for I:=1 to trunc(m) do
       begin
```

```
          r:=random;
          U:=U*r;
          end;
      x:=-media/m*ln(U);
end;
end.(unit distr)
Unit Prob;(Determina funcoes de probabilidade e os valores segundo as mesmas)
Interface
Uses Defvar, Dos,Distr;
     procedure escolhe_Funcao_probabilidade(Pont_job:ptr_job;Pont_tarefa:ponteiro;Indice_rota,Opcao:byte);
     procedure tempo_reparo(TN:ptr_celula; Indice:byte;var Tempo:real);
     procedure tempo_quebra(TN:ptr_celula; Indice:byte;var Tempo:real);
     procedure tamanho_de_lote(TN:ptr_job;var Tamanho:word);
     procedure tempo_de_chegada(TN:ptr_job;var Tempo:real);
Implementation
procedure escolhe_Funcao_probabilidade(Pont_job:ptr_job;Pont_tarefa:ponteiro;Indice_rota,Opcao:byte);
begin
     with Pont_job^.elemento[Indice_rota] do
     begin
     case distribuicao[Opcao] of
     0:Pont_tarefa^.T_ocupacao:=par_1[Opcao];
     1:dist_uniforme(par_1[opcao],par_2[Opcao],Pont_tarefa^.T_ocupacao);
     2:dist_exponencial(par_1[opcao],Pont_tarefa^.T_ocupacao);
     3:dist_gama(par_1[opcao],par_2[Opcao],Pont_tarefa^.T_ocupacao);
     4:dist_normal(par_1[opcao],par_2[Opcao],Pont_tarefa^.T_ocupacao);
     5:dist_b_neg(par_1[opcao],par_2[Opcao],Pont_tarefa^.T_ocupacao);
     6:dist_binomial(par_1[opcao],par_2[Opcao],Pont_tarefa^.T_ocupacao);
     7:dist_hiperg(par_1[opcao],par_2[Opcao],par_3[Opcao],Pont_tarefa^.T_ocupacao);
     8:dist_poisson(par_1[opcao],Pont_tarefa^.T_ocupacao);
     9:dist_m_Erlang(par_1[opcao],par_2[Opcao],Pont_tarefa^.T_ocupacao);
     end;
     end;
     with Pont_tarefa^ do
     begin
          if (Tam_lote > 1)
          then T_ocupacao:=Tam_lote*T_ocupacao+(Tam_lote-1)*T_set_up_lote;
     end;
end;
procedure tempo_reparo(TN:ptr_celula; Indice:byte;var Tempo:real);
begin
     with TN^.maquina[Indice] do
     begin
     case dist_reparo of
     0:Tempo:=par_rep1;
     1:dist_uniforme(par_rep1,par_rep2,Tempo);
     2:dist_exponencial(par_rep1,Tempo);
     3:dist_gama(par_rep1,par_rep2,Tempo);
     4:dist_normal(par_rep1,par_rep2,Tempo);
     5:dist_b_neg(par_rep1,par_rep2,Tempo);
     6:dist_binomial(par_rep1,par_rep2,Tempo);
     7:dist_hiperg(par_rep1,par_rep2,par_rep3,Tempo);
     8:dist_poisson(par_rep1,Tempo);
     9:dist_m_Erlang(par_rep1,par_rep2,Tempo);
     end;
     end;
end;
procedure tempo_quebra(TN:ptr_celula; Indice:byte;var Tempo:real);
begin
     with TN^.maquina[Indice] do
     begin
     case distr_quebra of
     0:Tempo:=par_quebra1;
     1:dist_uniforme(par_quebra1,par_quebra2,Tempo);
     2:dist_exponencial(par_quebra1,Tempo);
     3:dist_gama(par_quebra1,par_quebra2,Tempo);
     4:dist_normal(par_quebra1,par_quebra2,Tempo);
     5:dist_b_neg(par_quebra1,par_quebra2,Tempo);
     6:dist_binomial(par_quebra1,par_quebra2,Tempo);
     7:dist_hiperg(par_quebra1,par_quebra2,par_quebra3,Tempo);
     8:dist_poisson(par_quebra1,Tempo);
     9:dist_m_Erlang(par_quebra1,par_quebra2,Tempo);
     end;
     end;
end;
procedure tamanho_de_lote;
var
   T:real;
begin
     with TN^ do
     begin
```

```
        case distr_lote of
        0:T:=par_l1;
        1:dist_uniforme(par_l1,par_l2,T);
        2:dist_exponencial(par_l1,T);
        3:dist_gama(par_l1,par_l2,T);
        4:dist_normal(par_l1,par_l2,T);
        5:dist_b_neg(par_l1,par_l2,T);
        6:dist_binomial(par_l1,par_l2,T);
        7:dist_hiperg(par_l1,par_l2,par_l3,T);
        8:dist_poisson(par_l1,T);
        9:dist_m_Erlang(par_l1,par_l2,T);
        end;
        Tamanho:=trunc(T);
        end;
  end;
  procedure tempo_de_chegada;
  begin
       with TN^ do
       begin
       case distr_cheg_armaz of
       0:Tempo:=par_chegi;
       1:dist_uniforme(par_chegi,par_cheg2,Tempo);
       2:dist_exponencial(par_chegi,Tempo);
       3:dist_gama(par_chegi,par_cheg2,Tempo);
       4:dist_normal(par_chegi,par_cheg2,Tempo);
       5:dist_b_neg(par_chegi,par_cheg2,Tempo);
       6:dist_binomial(par_chegi,par_cheg2,Tempo);
       7:dist_hiperg(par_chegi,par_cheg2,par_cheg3,Tempo);
       8:dist_poisson(par_chegi,Tempo);
       9:dist_m_Erlang(par_chegi,par_cheg2,Tempo);
       end;
       end;
  end;
  end.
  Unit Comeca;
  Interface
  Uses Dos,Defvar,Lista_celula,Lista_evento,Lista_job,Lista_Tarefa,Prob;
       procedure N_grupos;
       procedure inicializacao;
       procedure primeiro_elemento(cabeca:ponteiro;P:ptr_evento);
       procedure inicio;
  Implementation
  procedure N_grupos;
  var
     G_celula:array [1..5] of byte;
     I,K:byte;
     TN:ptr_celula;
     Achei:boolean;
  begin
       for I:=1 to 5 do G_Celula[I]:=0;
       TN:=raiz_celula^.ptr_proximo;
       Achei:=true;
       Grupos:=0;
       while TN () nil do
       begin
            for I:=1 to TN^.N_maquinas do
            begin
                 K:=1;
                 while (K(=TN^.N_maquinas) and Achei do
                 begin
                      if (TN^.maquina[I].Ident_maquina () G_celula[K])
                      then begin
                           Grupos:=Grupos+1;
                           G_celula[K]:=TN^.maquina[I].Ident_maquina;
                           Achei:=false;
                           Num_grupo[TN^.maquina[I].Ident_maquina]:=1;
                           end
                      else Num_grupo[TN^.maquina[I].Ident_maquina]:=Num_grupo[TN^.maquina[I].Ident_maquina]+1;
                      K:=K+1;
                 end;
                 Achei:=true;
            end;
            TN:=TN^.ptr_proximo;
       end;
  end;
  procedure inicializacao;
  var
     Machine:byte;
     I:byte;
     TN1:ptr_celula;
```

```
    quebra:ptr_evento;
    T_quebra:real;
  begin
      Numero:=0;
      relogio:=0.0;
      Ind_Maq:=0;
      Final:=false;
      Maq:=0;
      Cel:=0;
      N_fila_ST:=0;
      N_grupos;
      TN1:=raiz_celula^.ptr_proximo;
      for I:=1 to Grupos do
      begin
          Condicao_Saturacao[I]:=0;
          Estado_saturacao[I]:=0;
      end;
      while (TN1 () nil) do
      begin
          TN1^.manip.situacao:=nao_ocupado;
          TN1^.manip.cond_operacao:=rotina;
          Output_Cel[TN1^.Ident_Celula]:=nil;
          for I:=1 to TN1^.N_maquinas do
          begin
              T_ocup_maq[TN1^.Ident_celula,I]:=0.0;
              T_reparo_maq[TN1^.Ident_celula,I]:=0.0;
              Machine:=TN1^.maquina[I].Ident_maquina;
              Condicao_Saturacao[Machine]:=Condicao_Saturacao[Machine]+TN1^.maquina[I].Cond_Saturacao;
              TN1^.maquina[I].situacao:=desocupado;
              TN1^.maquina[I].cond_operacao:=normal;
              Output_Maq[TN1^.Ident_Celula,I]:=nil;
              if TN1^.maquina[I].par_quebra1 () 0
              then begin
                  GetMem(quebra,SizeOf(registro_evento));
                  quebra^.operador:=quebra_maquina;
                  quebra^.Indice:=I;
                  quebra^.Maquina:=TN1^.maquina[I].Ident_maquina;
                  tempo_quebra(TN1,I,T_quebra);
                  quebra^.T_ocorrencia:=relogio+T_quebra;
                  quebra^.Cell:=TN1^.Ident_Celula;
                  insere_evento(raiz,quebra);
                  end;
              end;
          TN1:=TN1^.ptr_proximo;
          end;
      for I:=1 to N_equip_ST do
      begin
          ST[I].estado:=nao_ocupado;
          ST[I].cond_operacao:=rotina;
          end;
      T_ocupado_ST:=0.0;
      T_util_ST:=0.0;
      T_armaz_ST:=0.0;
      for I:=1 to Num_celulas do
      begin
          N_fila_Manipulador[I]:=0;
          T_ocupado_manip[I]:=0.0;
          T_ult_alt_fila_manip[I]:=0.0;
          Area_sob_curva_manip[I]:=0.0;
          T_util_manip[I]:=0.0;
          T_buffer_manip[I]:=0.0;
          end;
      for I:=1 to Grupos do
      begin
          N_fila_Maq[I]:=0;
          T_ult_alt_fila_maq[I]:=0.0;
          Area_sob_curva_maq[I]:=0.0;
      end;
      for I:=1 to Num_componentes do
      begin
          N_vezes_fila[I]:=0;
          T_demora_fila[I]:=0.0;
          N_jobs_completos[I]:=0;
      end;
end;
procedure primeiro_elemento(cabeca:ponteiro;P:ptr_evento);
var
   provisorio:ptr_evento;
   TK:ponteiro;
begin
```

```
      TK:=cabeca^.ptr_proximo;
      GetMem(provisorio,SizeOf(registro_evento));
      provisorio^.operador:=armazena;
      provisorio^.T_ocorrencia:=TK^.T_ordenacao;
      insere_evento(P,provisorio);
end;
procedure inicio;
begin
     new (raiz);
     raiz^.ptr_proximo:=nil;
     primeiro_elemento(armazenamento,raiz);
end;
end. (Fim da Unit)
{$O+,F+}
Unit Prior;
Interface
Uses Dos, Crt, Overlay, Letecla, Varler, Tela,defvar;
{$O Tela}
   procedure entr_sistema_prioridade;
Implementation
procedure entr_sistema_prioridade;
begin
     window(3,3,78,24);
     clrscr;
     window(1,1,80,25);
     resposta:= 'n';
     while(resposta='n') or (resposta='N') do
     begin
          tela_prioridade;
          sist_prioridade:=0;
          while((sist_prioridade(1) or (sist_prioridade)5)) do
          begin
               LeTeclado(30,20,1,'B',' ',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) then
                   sist_prioridade:=trunc(RealGenerico);
               gotoxy(30,20);
               write('     ');
               gotoxy(30,20);
               write(sist_prioridade);
          end;
          resposta:='';
          while((resposta()'N') and (resposta()'S')) do
          begin
               gotoxy(10,22);
               write('Esta tudo correto;        ');
               LeTeclado(29,22,1,'S','U',StrGenerica,RealGenerico,Resultado);
               if (Resultado=0) then
                   resposta:=Strgenerica;
          end;
     end;
end;
end.
{$O+,F+}
Unit menu;
Interface
Uses Dos, Crt, Overlay, tela, letecla, exec, dialogo;
     procedure menu;
Implementation
procedure menu;
var
   resposta:byte;
begin
     menu_principal;
     moldura2(1,1,80,25);
     LeTeclado(30,20,1,'S','U',StrGenerica,RealGenerico,Resultado);
     if Resultado=0 then
        resposta:=trunc(RealGenerico);
     Case resposta of
     1: begin
           entrada;
           clrscr;
           menu;
        end;
     2: begin
           execucao;
           clrscr;
           menu;
        end;
     3: begin
           clrscr;
```

```
            Halt;
        end;
     end;
end;
end.
{$O+,F+}
Unit Relatori;
Interface
uses overlay,dos,crt,varrer,varler,defvar,ler,tela,letecla;
             procedure relatorio(Nome_arq_grav_rel:string);
             procedure rel_info_gerais;
             procedure rel_celulas;
             procedure rel_coordenadas;
             procedure rel_componentes;
             procedure entr_nome_arq_rel;

Implementation
procedure rel_info_gerais;
var
  ci:byte;
begin
    WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
    WRITELN(ArqRelat,'NOME DO ARQUIVO DO SISTEMA: ',Nome_arquivo);
    WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
    if Nome_arquivo_coord()'' then
    begin
        WRITELN(ArqRelat,'NOME DO ARQUIVO DAS COORDENADAS: ',Nome_arquivo_coord);
        WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
    end;
    WRITELN(ArqRelat);
    WRITELN(ArqRelat,'                        INFORMACOES GERAIS');
    WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
    WRITELN(ArqRelat,'Nome da Simulacao: ',Nome_da_simulacao);
    WRITELN(ArqRelat);
    WRITELN(ArqRelat,'Tempo de Simulacao: ',T_simulacao:7:2);
    WRITELN(ArqRelat);
    WRITELN(ArqRelat,'Sistema de Prioridade: ',Sist_prioridade);
    WRITELN(ArqRelat);
    WRITELN(ArqRelat,'Numero de celulas: ',Num_Celulas);
    WRITELN(ArqRelat);
    WRITELN(ArqRelat,'Numero de Equipamentos do Sistema de Transferencia: ',N_equip_ST);
    for ci:=1 to N_equip_ST do
    begin
        WRITELN(ArqRelat);
        WRITELN(ArqRelat,'Nome do Sistema de Transferencia ',ci,': ',ST[ci].tipo);
        WRITELN(ArqRelat);
        WRITELN(ArqRelat,'Velocidade do Sistema de Transferencia ',ci,': ',ST[ci].velocidade:7:2);
    end;
    WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
end;
procedure rel_coordenadas;
var
  ci,c2:byte;
begin
    WRITELN(ArqRelat,'                          COORDENADAS DO SISTEMA');
    WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
    WRITELN(ArqRelat,dist_saida_x,dist_saida_y,' -) Coordenadas da saida do sistema');
    WRITELN(ArqRelat,dist_celula[0,1],dist_celula[0,2],' -) Coordenadas do Armazenamento');
    for ci:=1 to Num_celulas do
    begin
        WRITELN(ArqRelat,dist_celula[ci,1],dist_celula[ci,2],' (- Coordenadas da Celula ',ci);
    end;
    for ci:=1 to Num_celulas do
    begin
        for c2:=1 to Numero_maquinas[ci] do
        begin
            WRITE(ArqRelat,coord_maq_incelula[ci,c2,1],coord_maq_incelula[ci,c2,2]);
            WRITELN(ArqRelat,' (- Coordenadas da maquina ',c2,' da celula ',ci);
        end;
    end;
    WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
end;
procedure rel_celulas;
var
  ci,c2:byte;
begin
    WRITELN(ArqRelat,'                    DESCRICAO DAS CELULAS DO SISTEMA');
    rel_celula:=raiz_celula^.ptr_proximo;
    for ci:=1 to Num_celulas do
    begin
```

```
          with rel_celula^ do
          begin
               WRITELN(ArqRelat);
               WRITELN(ArqRelat);
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               WRITELN(ArqRelat,'        CELULA ',c1);
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               with manip do
               begin
                    WRITE(ArqRelat,'Nome do Manipulador: ',Nome);
                    WRITELN(ArqRelat,'      Velocidade do Manipulador: ',Veloc:7:2);
               end;
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               WRITELN(ArqRelat,'    DESCRICAO DAS MAQUINAS');
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               WRITELN(ArqRelat,'NOME                IDENT  T.SET.UP    C.SAT.  FPROB.Q.  PAR     FPROB.R    PAR');
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               for c2:=1 to Numero_maquinas[c1] do
               begin
                    with maquina[c2] do
                    begin
                         WRITE(ArqRelat,Name:20,' ',Ident_maquina:2,'     ',T_set_up:8:2,'     ');
                         WRITE(ArqRelat,Cond_saturacao:2,'          ',Distr_quebra:1,'     ');
                         acha_N_parametros(Distr_quebra,N_par_quebra);
                         WRITE(ArqRelat,par_quebra1:7:2,'     ');
                         WRITE(ArqRelat,Dist_reparo:2,'       ');
                         acha_N_parametros(Dist_reparo,N_par_reparo);
                         WRITE(ArqRelat,par_rep1:7:2,'  ');
                         WRITELN(ArqRelat);
                         if N_par_quebra>1 then
                         begin
                            WRITE(ArqRelat,'                                                    ');
                            WRITE(ArqRelat,par_quebra2:7:2,'  ');
                            if N_par_reparo>1 then
                            begin
                                 WRITE(ArqRelat,'                ');
                                 WRITELN(ArqRelat,par_rep2:7:2);
                            end
                            else
                                WRITELN(ArqRelat);
                         end;

                         if N_par_quebra>2 then
                         begin
                            WRITE(ArqRelat,'                                                    ');
                            WRITE(ArqRelat,par_quebra3:7:2,'  ');
                            if N_par_reparo>2 then
                            begin
                                 WRITE(ArqRelat,'                ');
                                 WRITELN(ArqRelat,par_rep3:7:2);
                            end
                            else
                               WRITELN(ArqRelat);
                         end;

                    end;
               end;
          end;
          rel_celula:=rel_celula^.ptr_proximo;
          WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
     end;
end;
procedure rel_componentes;
var
   c1,c2,c3:byte;
begin
     WRITELN(ArqRelat,'           DESCRICAO DOS COMPONENTES');
     rel_job:=raiz_job^.ptr_proximo;
     for c1:=1 to Num_componentes do
     begin
          WRITELN(ArqRelat);
          WRITELN(ArqRelat);
          WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
          WRITELN(ArqRelat,'    COMPONENTE ',c1);
          WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
          WRITELN(ArqRelat,'Nome: ',rel_job^.componente);
          WRITELN(ArqRelat,'Identificacao: ',rel_job^.identificacao);

          WRITE(ArqRelat,'F.Prob.Tempo Chegada: ',rel_job^.distr_cheg_armaz);
          WRITE(ArqRelat,'        Par 1: ',rel_job^.Par_cheg1);
```

```
          acha_N_Parametros(rel_job^.distr_cheg_armaz,N_par_cheg);
          if N_par_cheg)1 then
              WRITE(ArqRelat,'    Par 2: ',rel_job^.Par_cheg2);
          if N_par_cheg)3 then
              WRITE(ArqRelat,'    Par 3: ',rel_job^.Par_cheg3);
          WRITELN(ArqRelat);

          WRITE(ArqRelat,'F.Prob. Tamanho de lote: ',rel_job^.distr_lote);
          WRITE(ArqRelat,'     Par 1: ',rel_job^.Par_l1);
          acha_N_Parametros(rel_job^.distr_lote,N_par_Tam_lote);
          if N_par_Tam_lote)1 then
              WRITE(ArqRelat,'    Par 2: ',rel_job^.Par_l2);
          if N_par_Tam_lote)3 then
              WRITE(ArqRelat,'    Par 3: ',rel_job^.Par_l3);
          WRITELN(ArqRelat);
          WRITE(ArqRelat,'-----------------------------------------------');
          WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
          WRITELN(ArqRelat);
          WRITE(ArqRelat,'OPERACAO 1&Opcao F.PROB   P1       P2       P3     2&Opcao');
          WRITELN(ArqRelat,' F.PROB   P1       P2       P3     3&Opcao F.PROB   P1       P2       P3');
          WRITE(ArqRelat,'-----------------------------------------------');
          WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
          for c3:=1 to (rel_job^.last-1) do
          begin
              with rel_job^.elemento[c3] do
              begin
                  WRITE(ArqRelat,c3:2);
                  WRITE(ArqRelat,'          ');
                  for c2:=1 to 3 do
                  begin
                      WRITE(ArqRelat,maquina[c2]:2,'      ');
                      WRITE(ArqRelat,distribuicao[c2]:2,'    ');
                      acha_N_parametros(distribuicao[c2],N_par_prob);
                      WRITE(ArqRelat,Par_1[c2]:7:2,'   ');
                      if N_par_prob)1 then
                         WRITE(ArqRelat,Par_2[c2]:7:2,'  ')
                      else
                         WRITE(ArqRelat,'         ');
                      if N_par_prob)2 then
                         WRITE(ArqRelat,Par_3[c2]:7:2,'    ')
                      else
                         WRITE(ArqRelat,'           ');
                  end;
                  WRITELN(ArqRelat);
              end;
          end;
          rel_job:=rel_job^.ptr_proximo;
          WRITE(ArqRelat,'-----------------------------------------------');
          WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
      end;
end;
procedure relatorio(Nome_arq_grav_rel:string);
begin
     ASSIGN(ArqRelat,Nome_arq_grav_rel);
     REWRITE(ArqRelat);
     rel_info_gerais;
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     if Nome_arquivo_coord()'' then
     begin
         rel_coordenadas;
         WRITELN(ArqRelat);
         WRITELN(ArqRelat);
     end;
     WRITELN(ArqRelat);
     rel_celulas;
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     rel_componentes;
     CLOSE(ArqRelat);
end;
procedure entr_nome_arq_rel;
begin
     clrscr;
     gotoxy(15,1);
     WRITE('IMPRESSAO DOS RESULTADOS');
     moldura(2,2,79,24);
     resposta:='n';
     while (resposta='n') or (resposta='N') do
```

```
    begin
        primeira_tela_leitura;
        gotoxy(10,2);
        drive_arquivo:='';
        while(drive_arquivo='') do
        begin
            LeTeclado(29,3,30,'S','U',StrGenerica,RealGenerico,Resultado);
            if (Resultado=0) THEN
            drive_arquivo:=StrGenerica;
        end;
        arq_rel:='';
        while(arq_rel='') do
        begin
            LeTeclado(28,5,12,'S','U',StrGenerica,RealGenerico,Resultado);
            if (Resultado=0) THEN
            arq_rel:=StrGenerica;
        end;
        arq_rel:=drive_arquivo+'\'+arq_rel;
        gotoxy(3,3);
        write('                                                    ');
        gotoxy(28,5);
        write('          ');
        gotoxy(28,5);
        write(arq_rel);
        resposta:=' ';
        while((resposta()'N') and (resposta()'S')) do
        begin
            gotoxy(10,20);
            write('Esta tudo correto:        ');
            LeTeclado(29,20,1,'S','U',StrGenerica,RealGenerico,Resultado);
            if (Resultado=0) then
               resposta:=Strgenerica;
        end;
    end;
end;
end.
{$O+,F+}
Unit Relatres;
Interface
uses overlay,dos,crt,varrer,varler,defvar,ler,est;
            procedure relatresult(Nome_arq_grav_rel:string);
            procedure rel_res_celulas;
            procedure rel_res_sist_trans;
            procedure rel_res_componentes;
Implementation
procedure rel_res_celulas;
var
  ci,c2,N_cel:byte;
begin
    WRITELN(ArqRelat,'                ESTATISTICAS DAS CELULAS DO SISTEMA');
    rel_celula:=raiz_celula^.ptr_proximo;
    while (rel_celula () nil) do
    begin
        with rel_celula^ do
        begin
            N_cel:=ident_celula;
            Est_final_manip(N_Cel);
            WRITELN(ArqRelat);
            WRITELN(ArqRelat);
            WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
            WRITELN(ArqRelat,'        CELULA ',Ident_celula);
            WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
            WRITELN(ArqRelat,'NOME                  %OCUPACAO   %OCUP-UTIL   %OCUP-BUFFER    NUM MED FILA');
            WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
            with manip do
            begin
                WRITELN(ArqRelat,Nome:20,' | ',Perc_ocupado_manip:6:3,'       '
                        ,Perc_util_manip*100:6:3,'          ,Perc_buffer_manip*100:6:3);
            end;
            WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
            WRITELN(ArqRelat,'   Maquinas da Celula ',Ident_celula);
            WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
            WRITELN(ArqRelat,'NOME                IDENT     %OCUPACAO     %QUEBRA     ');
            WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
            for c2:=1 to N_maquinas do
            begin
                Est_final_maq(N_cel,c2);
                with maquina[c2] do
                begin
                    WRITELn(ArqRelat,Name:20,' |',Ident_maquina:2,'        ',Perc_ocupacao_maq:6:3,'       ',
```

```
                                        Perc_reparo_maq:6:3);
                    end;
                end;
            end;
            rel_celula:=rel_celula^.ptr_proximo;
            WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
        end;
        WRITELN(ArqRelat);
        WRITELN(ArqRelat);
        WRITELN(ArqRelat);
        WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
        WRITELN(ArqRelat,'                    ESTATISTICAS DOS GRUPOS DE MAQUINAS');
        WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
        WRITELN(ArqRelat,'GRUPO          Numero de maquinas          Numero medio em fila');
        WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
        For I:=1 to grupos do
        begin
            Est_final_grupos(I);
            WRITELN(ArqRelat,I:3,'                  ',Num_grupo[I]:3,'                          ',Num_medio_fila_maq:7:3);
        end;
        WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
    end;
    procedure rel_res_componentes;
    var
      ci,c2,c3:byte;
    begin
        WRITELN(ArqRelat,'          ESTATISTICAS DOS COMPONENTES');
        WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
        WRITELN(ArqRelat,'TAREFA                IDENT. * T.MED.ESPERA EM FILA * NUMERO DE TERMINADOS ');
        WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
        rel_job:=raiz_job^.ptr_proximo;
        for ci:=1 to Num_componentes do
        begin
            Est_final_comp(ci);
            WRITE(ArqRelat,rel_job^.componente:20,'    ');
            WRITEln(ArqRelat,rel_job^.identificacao,'          ',T_medio_espera:6:3,'            ',N_jobs_completos[ci]:7);
            WRITEln(ArqRelat,'-------------------------------------------------');
            rel_job:=rel_job^.ptr_proximo;
        end;
    end;
    procedure rel_res_sist_trans;
    var
      ci:byte;
    begin
            WRITELN(ArqRelat,'             ESTATISTICAS DOS SISTEMAS DE TRANSFERENCIA');
            WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
            WRITELN(ArqRelat,'NOME              %OCUPACAO   NUM.MED.FILA   %OPERACAO-CEL  %OPERACAO-ARMAZ');
            WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
            Est_final_ST;
        (for ci:=1 to N_equip_ST do begin)
            WRITELN(ArqRelat,ST[i].tipo,'              ',Perc_ocupado_ST*100:6:3,'       ',Num_medio_fila_ST:6:3,'           ',
                        Perc_util_ST:6:3,'           ',Perc_armaz_ST:6:3);
        (end;)
        WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
    end;
    procedure relatresult(Nome_arq_grav_rel:string);
    begin
        ASSIGN(ArqRelat,Nome_arq_grav_rel);
        APPEND(ArqRelat);
        WRITELN(ArqRelat);
        WRITELN(ArqRelat);
        WRITELN(ArqRelat);
        WRITELN(ArqRelat,'*************************************************************');
        WRITELN(ArqRelat,'********************** ESTATISTICAS **********************');
        WRITELN(ArqRelat,'*************************************************************');
        WRITELN(ArqRelat);
        WRITELN(ArqRelat);
        WRITELN(ArqRelat);
        rel_res_celulas;
        WRITELN(ArqRelat);
        WRITELN(ArqRelat);
        WRITELN(ArqRelat);
        rel_res_sist_trans;
        WRITELN(ArqRelat);
        WRITELN(ArqRelat);
        WRITELN(ArqRelat);
        rel_res_componentes;
        CLOSE(ArqRelat);
    end;
    end.
```

```
{$O+,F+}
Unit Relatres;
Interface
uses overlay,dos,crt,varrer,varler,defvar,ler,est;
             procedure relatresult(Nome_arq_grav_rel:string);
             procedure rel_res_celulas;
             procedure rel_res_sist_trans;
             procedure rel_res_componentes;
Implementation
procedure rel_res_celulas;
var
   c1,c2,N_cel:byte;
begin
     WRITELN(ArqRelat,'                ESTATISTICAS DAS CELULAS DO SISTEMA');
     rel_celula:=raiz_celula^.ptr_proximo;
     while (rel_celula <> nil) do
     begin
          with rel_celula^ do
          begin
               N_cel:=ident_celula;
               Est_final_manip(N_Cel);
               WRITELN(ArqRelat);
               WRITELN(ArqRelat);
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               WRITELN(ArqRelat,'         CELULA ',Ident_celula);
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               WRITELN(ArqRelat,'NOME                  %OCUPACAO   %OCUP-UTIL   %OCUP-BUFFER    NUM MED FILA');
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               with manip do
               begin
                    WRITELN(ArqRelat,Nome:20,' I ',Perc_ocupado_manip:6:3,'         '
                            ,Perc_util_manip*100:6:3,'          ',Perc_buffer_manip*100:6:3);
               end;
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               WRITELN(ArqRelat,'    Maquinas da Celula ',Ident_celula);
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               WRITELN(ArqRelat,'NOME                IDENT     %OCUPACAO     %QUEBRA     ');
               WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
               for c2:=1 to N_maquinas do
               begin
                    Est_final_maq(N_cel,c2);
                    with maquina[c2] do
                    begin
                         WRITELN(ArqRelat,Name:20,' I',Ident_maquina:2,'         ',Perc_ocupacao_maq:6:3,'         ',
                                 Perc_reparo_maq:6:3);
                    end;
               end;
          end;
          rel_celula:=rel_celula^.ptr_proximo;
          WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
     end;

     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
     WRITELN(ArqRelat,'                    ESTATISTICAS DOS GRUPOS DE MAQUINAS');
     WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
     WRITELN(ArqRelat,'GRUPO          Numero de maquinas          Numero medio em fila');
     WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
     For I:=1 to grupos do
     begin
          Est_final_grupos(I);
          WRITELN(ArqRelat,I:3,'                  ',Num_grupo[I]:3,'                          ',Num_medio_fila_maq:7:3);
     end;
     WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
end;
procedure rel_res_componentes;
var
   c1,c2,c3:byte;
begin
     WRITELN(ArqRelat,'           ESTATISTICAS DOS COMPONENTES');
     WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
     WRITELN(ArqRelat,'TAREFA               IDENT. * T.MED.ESPERA EM FILA * NUMERO DE TERMINADOS ');
     WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
     rel_job:=raiz_job^.ptr_proximo;
     for c1:=1 to Num_componentes do
     begin
          Est_final_comp(c1);
          WRITE(ArqRelat,rel_job^.componente:20,'     ');
```

```
          WRITELn(ArqRelat,rel_job^.identificacao,'          - ',T_medio_espera:6:3,'              ',N_jobs_completos[ci]:7);
          WRITELn(ArqRelat,'-------------------------------------------------');
          rel_job:=rel_job^.ptr_proximo;
      end;
end;
procedure rel_res_sist_trans;
var
   ci:byte;
begin
          WRITELN(ArqRelat,'              ESTATISTICAS DOS SISTEMAS DE TRANSFERENCIA');
          WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
          WRITELN(ArqRelat,'NOME              %OCUPACAO   NUM.MED.FILA   %OPERACAO-CEL  %OPERACAO-ARMAZ');
          WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
          Est_final_ST;
     (for ci:=1 to N_equip_ST do begin)
          WRITELN(ArqRelat,ST[i].tipo,'              ',Perc_ocupado_ST*100:6:3,'       ',Num_medio_fila_ST:6:3,'          ',
                    Perc_util_ST:6:3,'           ',Perc_armaz_ST:6:3);
     (end;)
     WRITELN(ArqRelat,'-------------------------------------------------------------------------------');
end;
procedure relatresult(Nome_arq_grav_rel:string);
begin
     ASSIGN(ArqRelat,Nome_arq_grav_rel);
     APPEND(ArqRelat);
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     WRITELN(ArqRelat,'*****************************************************************');
     WRITELN(ArqRelat,'********************** ESTATISTICAS **********************');
     WRITELN(ArqRelat,'*****************************************************************');
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     rel_res_celulas;
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     rel_res_sist_trans;
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     WRITELN(ArqRelat);
     rel_res_componentes;
     CLOSE(ArqRelat);
end;
end.
```

ARQUIVOS GERADOS PELO SISTEMA SIMFLEX

Arquivo de Definiçao de Entidades (exceto coordenadas)

```
Denis*(- Nome da Simulacao
2  (- Numero de celulas
1  (- Identificacao da celula 1
2  (- Numero de maquinas da celula 1
 1.00000000000000E+0000  (- Tempo de setup da maquina 1 da celula 1
maq1*  (- Nome da maquina 1 da celula 1
1  (- Identificacao da maquina 1 da celula 1
0 (- Condicao de saturacao da maquina 1
0  (- Funcao de probabilidade de quebra da maquina 1 da celula 1
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. de quebra da maquina 1
0  (- Funcao de probabilidade de reparo da maquina 1 da celula 1
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. de reparo da maquina 1
 1.00000000000000E+0000  (- Tempo de setup da maquina 2 da celula 1
maq12*  (- Nome da maquina 2 da celula 1
2  (- Identificacao da maquina 2 da celula 1
0 (- Condicao de saturacao da maquina 1
0  (- Funcao de probabilidade de quebra da maquina 2 da celula 1
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. de quebra da maquina 2
0  (- Funcao de probabilidade de reparo da maquina 2 da celula 1
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. de reparo da maquina 2
manip*  (- Nome do manipulador da celula 1
 0.0 (- Velocidade do manipulador da celula 1
2  (- Identificacao da celula 2
2  (- Numero de maquinas da celula 2
 1.00000000000000E+0000  (- Tempo de setup da maquina 1 da celula 2
maq21*  (- Nome da maquina 1 da celula 2
3  (- Identificacao da maquina 1 da celula 2
0 (- Condicao de saturacao da maquina 2
0  (- Funcao de probabilidade de quebra da maquina 1 da celula 2
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. de quebra da maquina 1
0  (- Funcao de probabilidade de reparo da maquina 1 da celula 2
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. de reparo da maquina 1
 1.00000000000000E+0000  (- Tempo de setup da maquina 2 da celula 2
maq22*  (- Nome da maquina 2 da celula 2
4  (- Identificacao da maquina 2 da celula 2
0 (- Condicao de saturacao da maquina 2
0  (- Funcao de probabilidade de quebra da maquina 2 da celula 2
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. de quebra da maquina 2
0  (- Funcao de probabilidade de reparo da maquina 2 da celula 2
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. de reparo da maquina 2
manip2*  (- Nome do manipulador da celula 2
 0.0 (- Velocidade do manipulador da celula 2
3  (- Numero de componentes
comp1*  (- Nome do componente 1
1 (- Identificacao do componente 1
2  (- Funcao prob. tempo chegada do componente 1
 5.00000000000000E-0001  (- Parametro 1 da Func. prob. tempo chegada do componente 1
0  (- Funcao prob. tamanho de lote do componente 1
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. tamanho de lote do componente 1
0 (- prioridade
3 (- Numero de operacoes do componente 1
1 (- Opcao 1 de maquina da operacao 1do componente 1
9 (- Funcao prob. opcao 1 da operacao 1 do componente 1
2 (- Opcao 2 de maquina da operacao 1 do componente 1
9 (- Funcao prob. opcao 2 da operacao 1 do componente 1
0 (- Opcao 3 de maquina da operacao 1 do componente 1
0 (- Funcao prob. opcao 3 da operacao 1 do componente 1
2 (- Numero de parametros da funcao da operacao 1 da opcao 1 do comp. 1
 2.00000000000000E+0000 (- Parametro 1.1
 1.00000000000000E+0000 (- Parametro 1.2
2 (- Numero de parametros da funcao da operacao 1 da opcao 2 do comp. 1
 2.00000000000000E+0000 (- Parametro 2.1
 2.00000000000000E+0000 (- Parametro 2.2
1 (- Numero de parametros da funcao da operacao 1 da opcao 3 do comp. 1
 0.00000000000000E+0000 (- Parametro 3.1
2 (- Opcao 1 de maquina da operacao 2do componente 1
9 (- Funcao prob. opcao 1 da operacao 2 do componente 1
0 (- Opcao 2 de maquina da operacao 2 do componente 1
0 (- Funcao prob. opcao 2 da operacao 2 do componente 1
```

```
 0 (- Opcao 3 de maquina da operacao 2 do componente 1
 0 (- Funcao prob. opcao 3 da operacao 2 do componente 1
 2 (- Numero de parametros da funcao da operacao 2 da opcao 1 do comp. 1
  2.00000000000000E+0000 (- Parametro 1.1
  2.00000000000000E+0000 (- Parametro 1.2
 1 (- Numero de parametros da funcao da operacao 2 da opcao 2 do comp. 1
  0.00000000000000E+0000 (- Parametro 2.1
 1 (- Numero de parametros da funcao da operacao 2 da opcao 3 do comp. 1
  0.00000000000000E+0000 (- Parametro 3.1
 3 (- Opcao 1 de maquina da operacao 3do componente 1
 9 (- Funcao prob. opcao 1 da operacao 3 do componente 1
 0 (- Opcao 2 de maquina da operacao 3 do componente 1
 0 (- Funcao prob. opcao 2 da operacao 3 do componente 1
 0 (- Opcao 3 de maquina da operacao 3 do componente 1
 0 (- Funcao prob. opcao 3 da operacao 3 do componente 1
 2 (- Numero de parametros da funcao da operacao 3 da opcao 1 do comp. 1
  2.00000000000000E+0000 (- Parametro 1.1
  3.00000000000000E+0000 (- Parametro 1.2
 1 (- Numero de parametros da funcao da operacao 3 da opcao 2 do comp. 1
  0.00000000000000E+0000 (- Parametro 2.1
 1 (- Numero de parametros da funcao da operacao 3 da opcao 3 do comp. 1
  0.00000000000000E+0000 (- Parametro 3.1
comp2*  (- Nome do componente 2
2 (- Identificacao do componente 2
2  (- Funcao prob. tempo chegada do componente 2
 5.00000000000000E-0001  (- Parametro 1 da Func. prob. tempo chegada do componente 2
0  (- Funcao prob. tamanho de lote do componente 2
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. tamanho de lote do componente 2
1 (- prioridade
3 (- Numero de operacoes do componente 2
3 (- Opcao 1 de maquina da operacao 1do componente 2
9 (- Funcao prob. opcao 1 da operacao 1 do componente 2
0 (- Opcao 2 de maquina da operacao 1 do componente 2
0 (- Funcao prob. opcao 2 da operacao 1 do componente 2
0 (- Opcao 3 de maquina da operacao 1 do componente 2
0 (- Funcao prob. opcao 3 da operacao 1 do componente 2
2 (- Numero de parametros da funcao da operacao 1 da opcao 1 do comp. 2
 2.00000000000000E+0000 (- Parametro 1.1
 1.00000000000000E+0000 (- Parametro 1.2
1 (- Numero de parametros da funcao da operacao 1 da opcao 2 do comp. 2
 0.00000000000000E+0000 (- Parametro 2.1
1 (- Numero de parametros da funcao da operacao 1 da opcao 3 do comp. 2
 0.00000000000000E+0000 (- Parametro 3.1
4 (- Opcao 1 de maquina da operacao 2do componente 2
9 (- Funcao prob. opcao 1 da operacao 2 do componente 2
0 (- Opcao 2 de maquina da operacao 2 do componente 2
0 (- Funcao prob. opcao 2 da operacao 2 do componente 2
0 (- Opcao 3 de maquina da operacao 2 do componente 2
0 (- Funcao prob. opcao 3 da operacao 2 do componente 2
2 (- Numero de parametros da funcao da operacao 2 da opcao 1 do comp. 2
 2.00000000000000E+0000 (- Parametro 1.1
 1.00000000000000E+0000 (- Parametro 1.2
1 (- Numero de parametros da funcao da operacao 2 da opcao 2 do comp. 2
 0.00000000000000E+0000 (- Parametro 2.1
1 (- Numero de parametros da funcao da operacao 2 da opcao 3 do comp. 2
 0.00000000000000E+0000 (- Parametro 3.1
3 (- Opcao 1 de maquina da operacao 3do componente 2
9 (- Funcao prob. opcao 1 da operacao 3 do componente 2
0 (- Opcao 2 de maquina da operacao 3 do componente 2
0 (- Funcao prob. opcao 2 da operacao 3 do componente 2
0 (- Opcao 3 de maquina da operacao 3 do componente 2
0 (- Funcao prob. opcao 3 da operacao 3 do componente 2
2 (- Numero de parametros da funcao da operacao 3 da opcao 1 do comp. 2
 2.00000000000000E+0000 (- Parametro 1.1
 1.00000000000000E+0000 (- Parametro 1.2
1 (- Numero de parametros da funcao da operacao 3 da opcao 2 do comp. 2
 0.00000000000000E+0000 (- Parametro 2.1
1 (- Numero de parametros da funcao da operacao 3 da opcao 3 do comp. 2
 0.00000000000000E+0000 (- Parametro 3.1
comp3*  (- Nome do componente 3
3 (- Identificacao do componente 3
2  (- Funcao prob. tempo chegada do componente 3
 0.50000000000000E+0000  (- Parametro 1 da Func. prob. tempo chegada do componente 3
0  (- Funcao prob. tamanho de lote do componente 3
 0.00000000000000E+0000  (- Parametro 1 da Func. prob. tamanho de lote do componente 3
1 (- prioridade
2 (- Numero de operacoes do componente 3
1 (- Opcao 1 de maquina da operacao 1do componente 3
9 (- Funcao prob. opcao 1 da operacao 1 do componente 3
0 (- Opcao 2 de maquina da operacao 1 do componente 3
```

```
 0 (- Funcao prob. opcao 2 da operacao 1 do componente 3
 0 (- Opcao 3 de maquina da operacao 1 do componente 3
 0 (- Funcao prob. opcao 3 da operacao 1 do componente 3
 2 (- Numero de parametros da funcao da operacao 1 da opcao 1 do comp. 3
  2.000000000000000E+0000 (- Parametro 1.1
  1.000000000000000E+0000 (- Parametro 1.2
 1 (- Numero de parametros da funcao da operacao 1 da opcao 2 do comp. 3
  0.000000000000000E+0000 (- Parametro 2.1
 1 (- Numero de parametros da funcao da operacao 1 da opcao 3 do comp. 3
  0.000000000000000E+0000 (- Parametro 3.1
 3 (- Opcao 1 de maquina da operacao 2do componente 3
 9 (- Funcao prob. opcao 1 da operacao 2 do componente 3
 4 (- Opcao 2 de maquina da operacao 2 do componente 3
 9 (- Funcao prob. opcao 2 da operacao 2 do componente 3
 0 (- Opcao 3 de maquina da operacao 2 do componente 3
 0 (- Funcao prob. opcao 3 da operacao 2 do componente 3
 2 (- Numero de parametros da funcao da operacao 2 da opcao 1 do comp. 3
  2.000000000000000E+0000 (- Parametro 1.1
  1.000000000000000E+0000 (- Parametro 1.2
 2 (- Numero de parametros da funcao da operacao 2 da opcao 2 do comp. 3
  2.000000000000000E+0000 (- Parametro 2.1
  2.000000000000000E+0000 (- Parametro 2.2
 1 (- Numero de parametros da funcao da operacao 2 da opcao 3 do comp. 3
  0.000000000000000E+0000 (- Parametro 3.1
 SIST1* (- Nome do sistema de transferencia
  0.0 (- Velocidade media do sistema de transferencia
 1 (- Numero de equipamentos do sistema de transferencia
```

Arquivo de Coordenadas

```
 4.000000000000000E+0001 0.000000000000000E+0001 (- Coordenadas da saida
 0.000000000000000E+0000 0.000000000000000E+0000 (- Coordenadas do armazenamento
-1.000000000000000E+0001 -1.000000000000000E+0001 (- Coordenadas da celula  1
 2.000000000000000E+0001 -2.000000000000000E+0001 (- Coordenadas da celula  2
 0.50000000000000
```

ANEXO C

RELATORIO EMITIDO PELO SISTEMA SIMFLEX

NOME DO ARQUIVO DO SISTEMA: C:\TURBO5\LSIMPLES\TESTE.DA

NOME DO ARQUIVO DAS COORDENADAS: C:\TURBO5\LSIMPLES\TESTE.CO

INFORMACOES GERAIS

Nome da Simulacao: DENIS

Tempo de Simulacao: 100.00

Sistema de Prioridade: 1

Numero de celulas: 2

Numero de Equipamentos do Sistema de Transferencia: 1

Nome do Sistema de Transferencia 1: SIST1

Velocidade do Sistema de Transferencia 1: 0.00

COORDENADAS DO SISTEMA

4.00000000000000E+0001 0.00000000000000E+0000 -) Coordenadas da saida do sistema
0.00000000000000E+0000 0.00000000000000E+0000 -) Coordenadas do Armazenamento
-1.00000000000000E+0001-1.00000000000000E+0001 (- Coordenadas da Celula 1
2.00000000000000E+0001-2.00000000000000E+0001 (- Coordenadas da Celula 2
5.00000000000000E+0000-5.00000000000000E+0000 (- Coordenadas da maquina 1 da celula 1
7.00000000000000E+0000-7.00000000000000E+0000 (- Coordenadas da maquina 2 da celula 1
-1.00000000000000E+0001-5.00000000000000E+0000 (- Coordenadas da maquina 1 da celula 2
5.00000000000000E+0000-7.00000000000000E+0000 (- Coordenadas da maquina 2 da celula 2

DESCRICAO DAS CELULAS DO SISTEMA

CELULA 1

Nome do Manipulador: manip Velocidade do Manipulador: 0.00

DESCRICAO DAS MAQUINAS

NOME	IDENT	T.SET.UP	C.SAT.	FPROB.Q.	PAR	FPROB.R	PAR
maq11	1	1.00	0	0	0.00	0	0.00
maq12	2	1.00	0	0	0.00	0	0.00

CELULA 2

Nome do Manipulador: manip2 Velocidade do Manipulador: 0.00

DESCRICAO DAS MAQUINAS

NOME	IDENT	T.SET.UP	C.SAT.	FPROB.Q.	PAR	FPROB.R	PAR
maq21	3	1.00	0	0	0.00	0	0.00
maq22	4	1.00	0	0	0.00	0	0.00

DESCRICAO DOS COMPONENTES

COMPONENTE 1

Nome: comp1
Identificacao: 1
F.Prob.Tempo Chegada: 2 Par 1: 5.00000000000000E-0001
F.Prob. Tamanho de lote: 0 Par 1: 0.00000000000000E+0000

OPERACAO	1&Opcao	F.PROB	P1	P2	P3	2&Opcao	F.PROB	P1	P2	P3	3&Opcao	F.PROB	P1	P2	P3
1	1	9	2.00	1.00		2	9	2.00	2.00		0	0	0.00		
2	2	9	2.00	2.00		0	0	0.00			0	0	0.00		
3	3	9	2.00	3.00		0	0	0.00			0	0	0.00		

COMPONENTE 2

Nome: comp2
Identificacao: 2
F.Prob.Tempo Chegada: 2 Par 1: 5.00000000000000E-0001
F.Prob. Tamanho de lote: 0 Par 1: 0.00000000000000E+0000

OPERACAO	1&Opcao	F.PROB	P1	P2	P3	2&Opcao	F.PROB	P1	P2	P3	3&Opcao	F.PROB	P1	P2	P3
1	3	9	2.00	1.00		0	0	0.00			0	0	0.00		
2	4	9	2.00	1.00		0	0	0.00			0	0	0.00		
3	3	9	2.00	1.00		0	0	0.00			0	0	0.00		

COMPONENTE 3

Nome: comp3
Identificacao: 3
F.Prob.Tempo Chegada: 2 Par 1: 5.00000000000000E-0001
F.Prob. Tamanho de lote: 0 Par 1: 0.00000000000000E+0000

OPERACAO	1&Opcao	F.PROB	P1	P2	P3	2&Opcao	F.PROB	P1	P2	P3	3&Opcao	F.PROB	P1	P2	P3
1	1	9	2.00	1.00		0	0	0.00			0	0	0.00		
2	3	9	2.00	1.00		4	9	2.00	2.00		0	0	0.00		

****
********** ESTATISTICAS **********
****

ESTATISTICAS DAS CELULAS DO SISTEMA

CELULA 1

NOME	%OCUPACAO	%OCUP-UTIL	%OCUP-BUFFER	NUM MED FILA
manip 1	0.000	0.000	0.000	

Maquinas da Celula 1

NOME	IDENT	%OCUPACAO	%QUEBRA
maq11	1	1.001	0.000
maq12	2	1.004	0.000

CELULA 2

NOME	%OCUPACAO	%OCUP-UTIL	%OCUP-BUFFER	NUM MED FILA

manip2	0.000	0.000	0.000

Maquinas da Celula 2

NOME	IDENT	%OCUPACAO	%QUEBRA
maq21	3	0.999	0.000
maq22	4	0.769	0.000

ESTATISTICAS DOS GRUPOS DE MAQUINAS

GRUPO	Numero de maquinas	Numero medio em fila
1	1	140.433
2	1	4.711
3	1	133.912
4	1	0.690

ESTATISTICAS DOS SISTEMAS DE TRANSFERENCIA

NOME	%OCUPACAO	NUM.MED.FILA	%OPERACAO-CEL	%OPERACAO-ARMAZ
SIST1	0.000	0.000	0.000	0.000

ESTATISTICAS DOS COMPONENTES

TAREFA	IDENT.	* T.MED.ESPERA EM FILA *	NUMERO DE TERMINADOS
comp1	1	18.706	9
comp2	2	16.328	11
comp3	3	10.340	25